VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

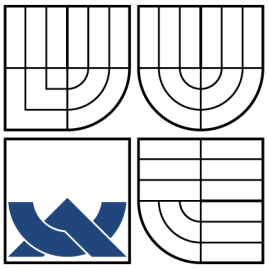
# DYNAMIC SCENE UNDERSTANDING FOR MOBILE ROBOT NAVIGATION

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

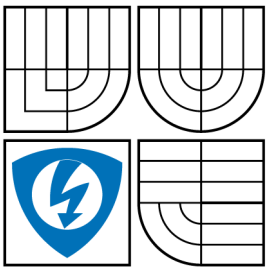
AUTHOR
AUTOR PRÁCE

Bc. ONDŘEJ MIKŠÍK

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND
COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

# DYNAMIC SCENE UNDERSTANDING FOR MOBILE ROBOT NAVIGATION

DYNAMICKÉ ROZPOZNÁVÁNÍ SCÉNY PRO NAVIGACI MOBILNÍHO ROBOTU

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE
Bc. ONDŘEJ MIKŠÍK

SUPERVISOR
VEDOUCÍ PRÁCE
doc. Ing. LUDĚK ŽALUD, Ph.D.

BRNO 2012

ZDE VLOŽIT LIST ZADÁNÍ

Z důvodu správného číslování stránek

## ABSTRACT

The thesis deals with dynamic scene understanding for mobile robot navigation. In the first part, we propose a novel approach to self-supervised learning – a fusion of frequency based vanishing point estimation and probabilistically based color segmentation. Detection of a vanishing point, is based on the estimation of a texture flow, produced by a bank of Gabor wavelets and a voting function. Next, the vanishing point defines the training area, which is used for self-supervised learning of color models. Finally, road patches are selected by measuring roadness score. A few rules deal with dark cast shadows, overexposed hightlights and adaptivity speed. In addition to that, the whole vanishing point estimation is refined – Gabor filters are approximated by Haar-like box functions, which enables efficient filtering via integral image trick. The tightest bottleneck, a voting scheme, is modified to coarse-to-fine, which provides a significant speed-up (more than $40\times$), while we loose only $3-5\%$ in precision.

The second part proposes a smoothing filter for spatio-temporal consistency of structured predictions, that are useful for more mature systems. The key part of proposed smoothing filter is a new, similarity metric, which is more discriminative than the standard Euclidean distance and can be used for various computer vision tasks. The smoothing filter first estimates optical flow to define a local neighborhood. This neighborhood is used for recursive averaging, based on the similarity metric. The total accuracy of proposed method measured on pixels with inconsistent labels between the raw and smooth predictions is almost $18\%$ higher than original predictions. Although, we have used SHIM, the algorithm can be combined with any other system for structured predictions (MRF/CRF, ...). The proposed smoothing filter represents a first step towards full inference.

## KEYWORDS

mobile robot, visual navigation, vanishing point, Gaussian Mixture Model, Matching Pursuit, scene understanding, spatio-temporal consistency

## ABSTRAKT

Diplomová práce se zabývá porozuměním dynamických scén pro navigaci mobilních robotů. V první části předkládáme nový přístup k "sebe-učícím" modelům – fůzi odhadu úběžníku cesty založeného na frekvenčním zpracování a pravděpodobnostních modelech pro segmentaci využívající barvu. Detekce úběžníku cesty je založena na odhadu dominantních orientací texturního toku, získáného pomocí banky Gaborových vlnek, a hlasování. Úběžník cesty poté definuje trénovací oblast, která se využívá k samostatnému učení barevných modelů. Nakonec, oblasti tvořící cestu jsou vybrány pomocí měření Mahalanobisovi vzdálenosti. Pár pravidel řeší situace, jako jsou mohutné stíny, přepaly a rychlost adaptivity. Kromě toho, celý odhad úběžníku cesty je přepracován – vlnky jsou nahrazeny aproximacemi pomocí binárních blokových funkcí, což umožňuje efektivní filtraci pomocí integrálních obrazů. Nejužší hrdlo celého algoritmu bylo samotné hlasování, proto překládáme schéma, které nejdříve provede hrubý odhad úběžníku a následně jej zpřesní, čímž dosáhneme výrazně vyšší rychlosti (až $40\times$), zatímco přesnost se zhorší pouze o $3 - 5\%$

V druhé části práce předkládáme vyhlazovací filtr pro prostorovo-časovou konzistentnost predikcí, která je důležitá pro vyspělé systémy. Klíčovou částí filtru je nová metrika měřící podobnost mezi třídami, která rozlišuje mnohem lépe než standardní Euclidovská vzdálenost. Tato metrika může být použita k nejrůznějším úlohám v počítačovém vidění. Vyhlazovací filtr nejdříve odhadne optický tok, aby definoval lokální okolí. Toto okolí je použito k rekurzivní filtraci založené na podobnostní metrice. Celková přesnost předkládané metody měřená na pixelech, které nemají shodné predikce mezi původními daty a vyfiltrovanými, je téměř o $18\%$ vyšší než u původních predikcí. Ačkoliv využíváme SHIM jako zdroj původních predikcí, algoritmus může být kombinován s kterýmkoliv jiným systémem (MRF, CRF, ...), který poskytne predikce ve formě pravěpodobností. Předkládaný filtr představuje první krok na cestě k plné usuzování.



## KLÍČOVÁ SLOVA

mobilní robot, vizuální navigace, úběžník cesty, směs Gaussovských modelů, Matching Pursuit, porozumění scéně, prostorově-časová konzistentnost

MIKŠÍK, Ondřej *Dynamic Scene Understanding for Mobile Robot Navigation*: master's thesis. Brno: Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Control and Instrumentation, 2012. 108 p. Supervised by doc. Ing. Luděk Žalud, Ph.D.

# DECLARATION

I declare that I have elaborated my master's thesis on the theme of "Dynamic Scene Understanding for Mobile Robot Navigation" independently, under the supervision of the master's thesis supervisor and with the use of technical literature and other sources of information which are all quoted in the thesis and detailed in the list of literature at the end of the thesis.

As the author of the master's thesis I furthermore declare that, concerning the creation of this master's thesis, master's thesis, I have not infringed any copyright. In particular, I have not unlawfully encroached on anyone's personal copyright and I am fully aware of the consequences in the case of breaking Regulation § 11 and the following of the Copyright Act No 121/2000 Vol., including the possible consequences of criminal law resulted from Regulation § 152 of Criminal Act No 140/1961 Vol.

Brno ..... .....

(author's signature)

# ACKNOWLEDGMENTS

I am very grateful to my supervisor Dr. Luděk Žalud for his advice, feedback and leadership. I would also like to thank my consultant prof. Martial Hebert, colleagues Daniel Munoz and Dr. Drew Bagnell as well as to all the other people who helped me with my internship in the Robotics Institute at Carnegie Mellon University.

My special thank goes to Petr Petyovsky, who spent a lot of his time with discussions with me on various topics as well as my other colleagues from Brno, prof. Pavel Jura and Dr. Miloslav Richter. Since the thesis summarizes my research over the last four years, I have to thank to my former advisor Dr. Krystian Mikolajczyk from the CVSSP at University of Surrey, who influenced me greatly, especially in my view on research.

Also, I cannot forget the fruitful discussions with Ang Li from Nanjing University, Dr. Jyrki Lahtonen from University of Turku, Dr. Hui Kong from MIT and all my friends and fellows students from Brno.

Last but not least, I am very grateful to all my family and my girlfriend Markéta for their support.

Brno ..... .....

(author's signature)

# CONTENTS

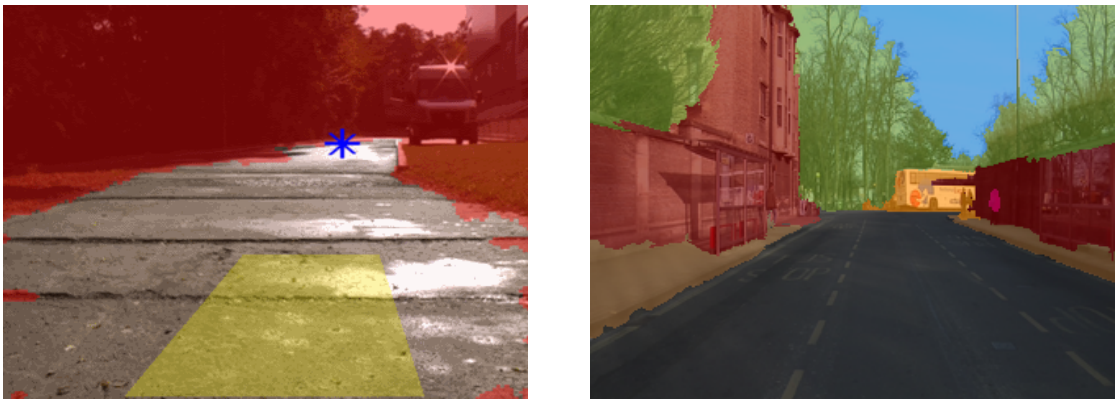
# LIST OF FIGURES

# LIST OF TABLES

# 1 INTRODUCTION

During the past few decades, the robotics community has made great efforts in developing autonomous or semi-autonomous robots. Such robots are able to perform desired tasks without continuous human guidance. One of the most fascinating problems for researchers working in the domain of mobile robotics is the development of a robot, which can autonomously operate in structured or unstructured environment. An ultimate goal perfectly represents a project of self-driving cars.

Many successful projects in the past have proven that the idea of a fully autonomous vehicle is not a utopia. Historically, one of the first attempts was Stanford Cart. Other famous projects are Argo [7, 5] and No Hands Across America [64]. The biggest boom started with the three challenges organized by Defense Advanced Research Projects Agency (DARPA). In the first DARPA Grand Challenge in 2004 (Mojave Desert), the best robot which was built by Carnegie Mellon University (CMU), only ran for 11.78km on length. Nevertheless it was a great achievement because they proved that it is possible to do.



(a) Self-supervised learning (b) Spatio-temporal consistency

Fig. 1.1: **Results of proposed systems** (best viewed in color).

One year later, all the robots exceeded the largest run from the previous competition and the winner was Stanley created by Stanford [86]. In 2007, the DARPA Urban Challenge took place in Victorville [84], so all the vehicles had to follow traffic regulations and the winner was Boss made by CMU [87]. Recently, a very impressive project was a trip from Parma to Shanghai connected with EXPO 2010 [8]. Today's most mature concept is likely the self-driving car being developed by Google [85].

While it might seem that all the projects are still far from perfect, there are technologies currently being developed that can be applied in various fields for army or space exploration robots. Afterwards, these technologies will enforce in our everyday life like safety systems in cars, mapping of abandoned mines, etc.

## 1.1 Problem Formulation

The general problem of mobile robot navigation can be summarized by three essential questions [6]:

- Where am I?
- Where am I going?
- How should I get there?

These questions are common for all robots regardless if the robot is mobile or not, or if it operates under water, on land, in the air, or in space. Answers for these questions are necessary for both the teleoperated as well as fully autonomous robots, however fully autonomous robots are able to find the answers themselves. The most interesting case is when robots are able to operate in an unknown environment.


(a) Orpheus-AC, BUT (b) Boss, Tartan Racing, CMU

Fig. 1.2: **Various robotic platforms**

Autonomous or semi-autonomous robots are complex systems consisting of many subsystems that deal with control, planning and perception. Autonomous robot navigation is one of the most extensively studied problem in the field of mobile robotics and reliable perception is crucial. The goal is to detect drivable surface ahead of the robot and plan the trajectory. This task is not easy even with the most advanced sensors. Usually, common sensors such as laser range finders provide information about obstacles in a near field, however long-range sensing is needed to be able to plan smooth trajectories for high speed vehicles. A combination of short-range sensors with a camera is commonly used to overcome such limitations.

In general, we can divide the algorithms addressing the problem of visual navigation into two groups. While the first one can be used with semi-autonomous robots

operating primarily in an unstructured environment, the latter can be used with more advanced systems that aim at fully autonomous behaviour.

We demonstrate the demands on algorithms for the first group on the Orpheus-AC reconnaissance mobile robot, which is being developed by Brno University of Technology for the Czech Army. Its primary task is to make the measurement and identification in areas with the highest risk of massive contamination.

The robot is primarily teleoperated. However, since it may move rapidly (about 15 km/h) ahead of the accompanying vehicle even in relatively hard terrain, it may be difficult, or even impossible, for the operator to directly control the robot in the moving vehicle in some situations. For this reason, it might be useful to have a system that would be able to automatically control the robot's movement in order to follow the road. Several important features and demands of the system come from the description of the mission – it should be able to:

- operate under a wide spectrum of operational conditions regarding climate, and surrounding environment – the system has to reliably find the way in diverse light conditions, like in direct sunlight, overcast, ...
- reliably drive on both high-quality roads as well as on roads barely visible even for humans including sand, concrete, tarmac, gravel, etc.
- use a minimum number of sensors – since the robot is intended to work in contaminated areas, it has to be extremely easy-to-decontaminate. Every irregularity on robot's surface means a serious problem. The robot is teleoperated, so it is already equipped with a high quality camera, which is an obvious source of data.

In addition to the above mentioned demands, it is obvious that an easy-to-use system is needed that does not require any difficult training or calibration, since the time-to-deployment is critical.

On the other hand, more advanced projects aiming at fully autonomous behaviour like Google self-driving cars [85] can use many other sensors and vision is just one of them. Because such systems usually work in very specific environments (city, highway, etc.), it is possible to use previously trained models. On the other hand, the system should provide more reliable output and give information about other objects (i.e. cars, pedestrians, buildings, ...), so that the planning algorithms would benefit from it. Thus, the total scene understanding can be seen as a reasonable prerequisite or subsystem for such systems.

## 1.2 Challenges

Dynamic scene understanding for navigation of mobile robots is a complex problem where various sensors can be used, including ultrasound sensors, radars, or thermal cameras. The importance of laser range finders and cameras has grown during the past few years, since they provide more information about the environment. Although the interpretation of measured data is not trivial, recent advances in data processing are very promising. It is difficult to say which sensor is the best, given that each has advantages and drawbacks. We have decided to use cameras since they are suitable for long-range sensing and almost every robot is already equipped with one. Even humans receive most information by vision.

However, dynamic scene understanding for visual navigation of mobile robots is a challenging task for various reasons. Even if we passover all the troubles related with image acquisition process, there still exist many reasons why it is quite difficult to recognize *objects* (cars, pedestrians, ...) and *stuff* (sky, ground, ...) in video sequences. Let us summarize some of them:

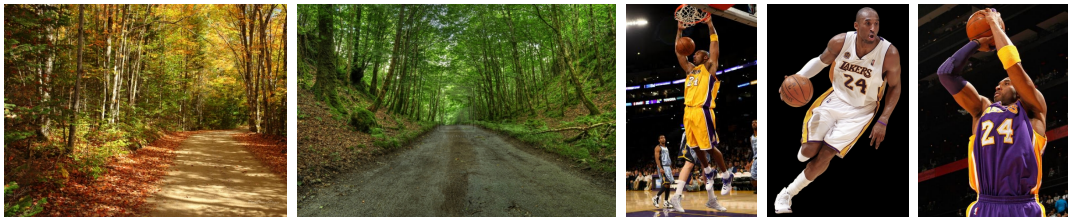
- **Viewpoint changes** – objects are seen under different viewpoints. It is absolutely necessary to deal with rotation, translation, scale, and affine invariant.
- **Intra-class variability** – objects from the same semantic class have various appearance (e.g. ground, sand, concrete, tarmac, ...)
- **Various illumination conditions** – objects or their fragments are seen under different illumination throughout the day, even at the same time, due to shadows or overexposed parts of a scene.
- **Deformability and poses** – objects and stuff look completely different even if they are seen from the same viewpoint (e.g. human body poses, ...)

Classical, early approaches to the computer vision that attempted to draw some simple recipes consisting of steps like blurring, edge detections, thresholdings, morphological operators and others, that are controlled in every single step by various "magic numbers" are doomed to fail – although they work well on a very small subset of training images, the real ones are much more complex.

Fortunately, we have two powerful tools that are able to deal with these challenges – a huge amount of data and models that are able to capture their properties and relationships among them. Perhaps, the most commercially successful application, which extensively use both, is the Microsoft Kinect. The rest of the thesis aims to address such models that are able to understand dynamic scenes for mobile robots.

(a) Viewpoint change – Hamerschlag Hall, Carnegie Mellon University

(b) Intra-class variability – TerraMax, Crusher, Shelley



(c) Various illumination conditions

(d) Human poses

Fig. 1.3: **Challenges in computer vision**, courtesy of [61]

## 1.3 Applications

Although we are motivated by robot navigation in terms of autonomous driving, it is possible to use it for the following applications (especially the second part of the thesis):

- **Perception and grasping interaction** – robot needs to recognize the object to be able to grasp it, object needs to be moved somehow to be recognized.
- **Video editing** – time consuming manual editing of videos (e.g. remove some object, ...) captured in natural environments can be improved with manual editing of just few frames and labels propagation.
- **Human-machine interaction** – improvements of HMI (advanced interactions, not just recognising of faces, ...) are possible with scene understanding.
- **Extended reality** – would benefit from knowledge of relationships between objects and scene context to deal with occlusions, ...

## 1.4 Thesis Structure

The rest of the thesis is organized as follows: in chapter 2, a brief review of state-of-the-art methods and computer vision background is given. Then, we propose a novel method based on self-supervised learning suitable for visual navigation of robots such as Orpheus-AC in chapter 3. Chapter 4 addresses the problem of spatio-temporal consistency for dynamic scene understanding suitable for more advanced systems and proposes a novel, similarity metric. The results are discussed in chapter 5. Appendix A summarizes naive approaches to spatio-temporal consistency and appendix B describes contents of the enclosed DVD.

## 1.5 Videos & Papers

The following papers about the presented research have been published so far:

(1) **Miksik O.**, Mikolajczyk K.: Local Detectors and Descriptors for Fast Feature Matching. Under review – *International Conference on Pattern Recognition (ICPR)* 2012

(2) **Miksik O.**: Rapid Vanishing Point Estimation for General Road Detection. In *International Conference on Robotics and Automation (ICRA)*, St. Paul, USA 2012

(3) **Miksik O.**, Petyovsky P., Zalud L., Jura P.: Robust Detection of Shady and Highlighted Roads for Monocular Camera Based Navigation of UGV. In *International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011

(4) Richter M., Petyovsky P., **Miksik O.**: Adapting Polynomial Mahalanobis Distance for Self-supervised Learning in an Outdoor Environment. In *International Conference on Machine Learning and Applications (ICMLA)*, Honolulu, USA, 2011

(5) **Miksik O.**: Road Detection in an Outdoor Environment. In *Student EEICT*, Brno, Czech Republic, 2011

More papers about the most recent parts of the thesis are coming soon.

Since we are interested in dynamic scenes understanding, it is difficult to show all the results as static images printed on a paper. Thus, we encourage the reader to see the accompanied videos on the enclosed DVD or `http://www.miksik.co.uk`

# 2 STATE OF THE ART

*This chapter discusses state-of-the-art methods. First, we give a brief overview of the most important systems for visual navigation developed throughout the years in section 2.1. Next, we discuss various approaches addressing scene understanding in section 2.2. Section 2.3 compares the proposed methods with the most related state-of-the-art approaches and section 2.4 gives a brief overview of various computer vision and machine learning techniques that we use throughout the thesis.*

## 2.1 Vision-based Navigation Systems

Many papers about vision-based road segmentation have been published during the last two decades. Most of the early systems have focused on structured roads. A well known project was developed in Carnegie Mellon University (CMU)'s Navlab [14, 15] that uses a number of Gaussian color models to represent the road and non-road colors (*UNSCARF*, *SCARF*). This navigation system is considered powerful because it deals with both intersections and shadows, however, it requires some overlapping between the frames. Thus, this system is not convenient for suddenly changing road surfaces. A similar project based on stereo vision is named *ARGO* from Università di Parma [7, 5]. Another project from CMU Navlab called *ALVINN* deals with both structured and unstructured roads. Nonetheless an artificial neural network classifier is used, which means that it requires previously learned road models [63].

Estimation of an optical flow [48, 90] is very popular, especially for visual odometry, however such approaches fail on chaotic roads when the camera is unstable and the optical flow estimation is not sufficiently robust. Other methods attempt to use Hough transform [9], or radar [53]. The main drawback of these methods are that they provide good performance only for roads with noticeable marking or borders.

Dahlkamp et al. [17] proposed an algorithm for self-supervised learning based on a combination of laser range finders and probabilistic color models represented by a Gaussian Mixture Model (GMM). Efficiency of the algorithm, which was used in Stanford's vehicle *Stanley* was proven by winning Darpa Grand Challenge [83]. Dong-Si et al. [18] proposed a modification to this algorithm – substituting laser range finders with stereo vision. A closely related paper was published by Gordic and Mulligen [23], who introduced a novel, more discriminative metric for measuring distances in a high dimensional and non-linear space, called Polynomial Mahalanobis distance (PMD).

Another branch of research is represented by approaches that estimate the so-called *Vanishing Point* (VP) of the road. The original paper by Rasmussen [66] investigates the grouping of dominant orientations of a texture flow that is suitable for unstructured, or ill-structured roads with no significant borders. The algorithm consists of two stages: (1) estimation of dominant orientations by a bank of Gabor wavelets [44], and (2) a voting scheme, which is used to determine the most likely coordinates of the road's vanishing point.

Rasmussen's approach works well in an outdoor environment with ill-structured roads which are barely visible even for humans. The above mentioned algorithm does not require any a priori knowledge about the road surface, difficult classifier training, etc. It provides information about the correct course for robot navigation, however the main drawbacks are the lack of information about the free space ahead of the robot and computational complexity. The later refinement employs laser range finders to deal with obstacles [67].

Another paper by Kong et al. [39, 38] proposed the idea of a locally adaptive soft voting scheme to prevent tending to favour points that are high in the image, which sometimes leads to large errors in the estimation of the vanishing point. The second important part of these papers discuss road segmentation by an Orientation Consistency Ratio and the two most dominant edges.

Finally, an approach published by Qi et al. [92] is similar - an example-based global image matching method is used to get an approximate idea of clear path candidate regions, and a Gaussian Mixture Model models local image patches to further improve the clear path detection.

## 2.2 Scene Understanding

All above mentioned mentioned approaches, aim at the extraction of road and non-road regions only. Such systems are very useful for semi-autonomous robots (e.g. rescue robots in the case of signal loss [57, 95, 96]) or autonomous vehicles in nature (e.g. desert [68, 67, 83], snowy roads [9], ...), however do not allow any advanced planning since the system does not have any knowledge about objects and stuff in a scene or relationships between them.

Semantic scene understanding was one of the first grand goals in computer vision in late 1970s, however, early approaches had to face many issues [27]. Let us mention

the lack of computational resources for all of them, which led to the extensive use of heuristics. Consequently, none of them were particularly successful and this line of research had been less and less intensively studied over the years because researchers had started to doubt about the goal.

During the past decade, big advancements in learning methods were proposed, which allow to learn relationships between the small image patches and even between the objects themselves. A big boom has started with Conditional Random Fields (CRF) proposed by Lafferty et al. [43] for natural language processing, which offer several advantages over Hidden Markov Models (HMMs) and stochastic grammars for such tasks, including the ability to relax strong Markovian independence assumptions made in those models.

A Conditional Random Field (CRF) can be viewed as a random field globally conditioned on the data. In the domain of computer vision, CRFs were first used by Kumar and Hebert [40] for the modelling of direct relationships between objects and for the detection of man-made structures. Many improvements to the CRF framework were proposed during the years, including two layered CRF to handle multiple classes and multi-layered CRF to encode both the short-range interactions (e.g. pixel-wise label smoothing) as well as the long-range interactions (e.g. relative configurations of objects or regions) [28, 41] or higher order potentials which improve labeling around object boundaries [35] or object detectors [77].

The other approaches that can be mentioned model relationships between regions, scenes, etc. Remarkable high level systems that put objects in perspective were developed by Hoiem et al. [30, 29]; Gupta et al. attempt to improve the scene labeling quality by discarding physically implausible environments by physical constraints [25].

Although the conditional random fields have proven to be a powerful tool in many computer vision problems, the exact inference is considered to be NP-hard and intractable [36]. A big effort was done in the field of an approximate inference, however learned models are tightly tied to the chosen inference procedure [34]. A great survey of probabilistic models for computer vision can be found in a recently published a book by Prince [65] or book by Koller and Friedman [37].

More recently, Munoz et al. [59] propose an efficient alternative to the above mentioned probabilistic graphical models to overcome such limitations. Their approach starts with an over-segmentation of an image. Then, the classifiers are sequentially

trained in coarse-to-fine manner to predict the label distributions. A first classifier is trained at the top level in the hierarchy to classify the top level's regions. Next, a second classifier is again trained over the level's regions but now also uses the first classifier's predictions from neighboring regions to encode context. The output of these classifiers is iteratively used in child's regions with child's features to get finer labeling. This procedure is repeated until the bottom leaves are reached. This algorithm has proven to be very efficient in both terms, a precision and computational complexity. It should be noted, that hierarchical inference machines are not constraint to 2D images only, Xiong et al. [94] propose a refinement which addresses the problem of scenes understanding and point-wise assignment of semantic labels from 3-D laser scans.

The main problem of all mentioned approaches is, that they exploit only spatial information, however none of them investigates temporal consistency across the multiple frames of a video stream, which means that the predictions tend to flicker and do not provide stable output (see Fig. 4.3) which is crucial for robotics (and many other) applications. One of the first approaches exploiting temporal consistency was proposed by Brostow et al. [11] utilize structure from ego-motion technique to automatically generate a 3D point clouds from video sequences, that are projected to the 2D image and used in decision forest classifier to perform a coherent semantic segmentation. Wojek and Schiele [91] proposed to use a dynamic conditional random field combined with an object detector and model the object dynamic as a flock of extended Kalman filters (EKF).

Ess et al. [19] build on the previous approach, however their application aims at inner-city scene understanding and road types classification (e.g. crossing, ...) with temporal smoothing based on a single Hidden Markov Model (HMM) for road type and one independent HMM for each object. Sturgess et al. [76] extends the motion feature set from [11] by appearance based features (HoG, textons, color, location, ...) and employ robust higher order $P^n$ potentials and object detectors [77].

Xiao and Quan [93] propose similar approach to [11], however, they use graph-based (Markov Random Field defined on superpixels) to enforce consistency of the segmentation result across multiple views.

A closely connected line of research is the domain of spatio-temporal segmentation for tracking or recognizing human actions in a video stream. For all of them, we mention Grundmann et al. [24], who extend spatial segmenter of Felzenszwalb and Huttenlocher [20] into time dimension by adding edges to the graph between regions

in a time window. The weights of edges correspond to a $\chi^2$ distance between region descriptors, in particular *Lab* histograms. Such approaches have very high memory requirements, which is overcome by a clip-based processing, which can be seen as an analogy to the overlap-add method for the short time Fourier transform computation. The segmentation results can be improved by the incorporation of an external dense optical flow. This concept was further investigated by Lezama et al. [47], who addressed the problem of region merging from two different classes by point tracks.

## 2.3 Comparison with the State of the Art

This section summarizes the differences between the most related state-of-the-art methods and our approach. First, we compare the proposed self-supervised learning algorithm with approaches, which have inspired and influenced us. Then, we discuss the differences between our method and the state-of-the-art approaches for spatio-temporal consistency for dynamic scene understanding.

### 2.3.1 Adaptive Road Extraction

The proposed self-supervised learning algorithm is mostly influenced by approaches used during the DARPA Grand Challenge 2005. Although, both keyparts of our algorithm – vanishing point estimation and texture segmentation have been refined later on, the main ideas remain the same.

#### Dahlkamp et al., RSS 2005

Our method presented in chapter 3 is mainly influenced and inspired by a self-supervised learning algorithm, which was successfully used in *Stanley* [75], during the Darpa Grand Challenge 2005 [83]. The main difference in our approach is, that Dahlkamp et al. use laser range finders to detect the training area while we use estimated vanishing point. This is motivated by the fact, that computer vision is used primarily for long-range sensing, which allows them high speed driving (it was reported, that long-range sensing was the key part for finishing the race with the best time), while short-range sensing and obstacle avoidance is done by other sensor.

The consequences are obvious – they do not need to store many models from history, because if some model is missing due to the changing surface for a few frames until it is adaptively learned, it is possible to use lidars. In contrast to [17], we do not use Expectation-Maximization (EM) algorithm for training of GMM since k-means clustering provides almost similar covariance matrices and is computationally

cheaper. Another benefit of k-means clustering is that we do not need to determine the number of clusters manually, however we can use hierarchical agglomerative (bottom-up) clustering. Since Dahlkamp et al. [17] used a relatively small number of models in their approach, their models are relatively quickly discarded if they do not correspond to the current surface, however in our case, we need to store more models to avoid frequent learning from scratch. This means that we face the windup effect which influences the speed of adaptivity, because GMM update rules have an integral character. Finally, they do not address the question of dealing with challenging illumination conditions such as overexposed highlights or dark cast shadows.

**Dong-Si et al., IROS 2008**

The most similar approach to our algorithm was proposed by Dong-Si et al. [18] who extended Dahlkamp et al. [17]. We construct and update GMM in the same manner as Dong-Si et al. [18], although they mention decay factor whose details are not discussed in the paper. The main difference is in extraction of the training area – Dong-Si et al. [18] perform 3D reconstruction to determine the training area. Only objects with height more than 10cm are considered as obstacles, which means that this approach may fail if there are not any high borders between the road and non-road regions (e.g. road vs. grass). We address the problem of loosing of learned GMM models if the robot is among shadows or overexposed highlights in the same manner as Dong-Si et al. [18]. In addition to that, we also use "preprocessors" similar to Rauskolb et al. [69] to label pixels which cannot be classified since they are too dark or too bright.

**Rasmussen, CVPR 2004**

So far, we have discussed only the road segmentation algorithm. As we have already mentioned, our training area is determined by the vanishing point. The most influencing work was proposed by Rasmussen [66, 68, 67]. This approach consists texture flow estimation and voting. In contrast to Rasmussen, we use decomposed multiscale Gabor wavelets (5 scales) into linear combinations of Haar-like binary box functions [81], that allows fast filtering via integral image trick [88]. Our voting scheme use locally adapted soft-voting strategy with efficient computation via superpixels, which does not tend to favouritism of voting candidates higher in the image and is up to 50× faster than Rasmussen's global voting strategy with hard assignment.

#### Kong et al., CVPR 2009, T-IP 2010

Kong et al. [39, 38] proposed several improvements to Rasmussen's approach which we have adopted. Extension to Rasmussen's work are multiscale approach to texture flow estimation and using of confidence score instead of maximum response for assignment of dominant orientation. By comparison with our approach, the main difference is that Kong et al. use FFT for measuring of responses while we use Haar-like box functions [81] and integral image trick [88]. Moreover, we refined the evaluation of the confidence score, since the formula proposed by Kong et al. allows sharp maxima, however does not reject responses with multiple maxima, which is not sufficient for reliable estimation.

Kong et al. [39, 38] also proposed a locally adapted soft-voting scheme, which reduces the number of voting pixels and penalizes the weights of pixels far away from the voting candidate. We have adopted all these modifications, however in addition to that, we introduce superpixels and propose coarse-to-fine voting, which is up to 40× faster than Kong et al., while the precision is comparable.

### 2.3.2 Dynamic Scene Understanding

Next, we discuss the similarities of our approach with the most related works which investigate spatio-temporal consistency. We have decided to propose a temporal smoothing filter for structured predictions instead of modifying some particular approach to full inference. The main advantage of this solution is that it reduces the flickering effects and can be easily combined with various approaches to pixel-wise labeling.

#### Motion Features

The first subset of methods employ motion features proposed by Brostow et al. [11]. These features (height above the camera, closest distance to camera path, surface orientation, track density, backprojection residual) are extracted from automatically generated 3D point clouds by structure from motion and are projected from the 3D world onto the 2D image plane and clustered using the K-means algorithm. Sturgess et al. [76] achieve a better results with combination of motion features and appearance feature. We do not use these motion features since it require an extra setup for the capturing system; the only information about the motion are velocity vectors extracted by large displacement optical flow [12].

### Graph-based approaches

Wojek and Schiele [91] propose to us a dynamic Conditional Random Field with directed links from frame $t$ to $t+1$, that model the dynamics of scene and objects. Since both are moving independently, Extended Kalman Filters (EKFs) are used for the tracking of objects, and scene dynamics is directly propagated from the previous frame. Similar approach was proposed by Ess et al. [19], who attempted to use HMMs to model the transitions between road types and objects. Our approach does not make any difference between objects (cars, pedestrian, ...) and stuff (sky, grass, ...). Although it is reasonable to use different models for temporal consistency of objects and stuff, the problem is that such approach requires reliable object detection and is much closer to tracking than smoothing.

### Xiao and Quan, ICCV 2009

Xiao and Quan [93] propose a crossover between the methods based on the 3D point clouds and graph-based methods. In contrast to Brostow et al. [11] and Sturgess et al. [76], they use different motion features, which do not require camera calibration. The 2D features are extracted on superpixels and consist of a 192 dimensional vector, which contains bunch of statistics (the median, deviation, skewness, kurtosis) of the *Lab* and *RGB* color space components, and a filter bank, which is made of three Gaussians, four Laplacians of Gaussians and four first-order derivatives of Gaussians. Unlike to us, this approach obviously expects environment which can be described as a "Manhattan" world and the processing of each sequence breaks down at the turn in the driving path.

### Munoz et al., ECCV 2010, Xiong et al., ICRA 2011

An efficient alternative to the MRF/CRF based systems were proposed by Munoz et al. [59], and its extension to the 3D point clouds by Xiong et al. [94], respectively. Although the Stacked Hierarchical Inference Machine (SHIM) is considered to be one of the best among the state-of-the-art methods, all labels are independently predicted at each frame, which tends to the flicker effects and the predictions are unstable over the time. Our method does not modify the inference process, however proposes a temporal smoothing filter, which reduce the most flickers. An important advantage is that we do not make any assumptions specific to this system. Hence, we can combine our smoothing filter with other approaches.

## 2.4 Computer Vision & Machine Learning Background

So far, we have described the related approaches. In this section, we give a brief overview of commonly used computer vision methods, that are useful for the proposed algorithms. We discuss instance-based learning methods in section 2.4.1, optical flow estimation in section 2.4.2 and features used for metric learning 2.4.3. Finally, we briefly mention FH-segmentation in section 2.4.4. Refer the books for more comprehensive overview.

### 2.4.1 Instance-based Learning

The grouping of similar features into clusters is addressed by unsupervised learning algorithms such as k-means clustering or Expectation Maximization (EM). Given a dataset of $N$ features $\mathbf{x}_i$ and a number of desired clusters $k$, where $k < N$, the goal of k-means clustering is to iteratively assign data vectors to the $k$ centers and update each of the $k$ centers to the mean of the data vectors assigned to it, until the algorithm converges and a local minimum of the criterion is achieved

$$J = \sum_{j=1}^{k} \sum_{\mathbf{x}_i \in C_j} \|\mathbf{x}_i - \mu_j\|^2 \tag{2.1}$$

where $\mu_j$ is the mean over all points in cluster $C_j$. Practically speaking, (0) $k$ centers are randomly initialized, (1) each feature vector $\mathbf{x}_i$ is assigned to the cluster with closest mean $\mu_j$ at each step and (2) the new mean for each cluster is computed. Steps (1) and (2) are iteratively performed until the algorithm converges.

Expectation-Maximization (EM) can be seen as a generalization of k-means (although EM is a Maximum Likelihood Estimator (MLE) in general). Given a set of observed data $\mathbf{X}$, a set of unobserved latent data $\mathbf{Z}$ and a vector of unknown parameters $\theta$, EM iteratively (0) initialize the parameters of $\theta$ to some random values, (1) compute the best values of $\mathbf{Z}$ given these parameter values and (3) use the computed values of $\mathbf{Z}$ to find better estimates for $\theta$. Step (1) and (2) are iteratively performed until the algorithm converges.

The main difference between k-means and EM is, that k-means makes *hard assignments* of data $\mathbf{x}_i$ to clusters, while EM makes *soft assignments*. Hence, the transition of clusters from initialization state to the final state is much more smooth with EM, while the convergence of k-means is usually faster. More details and other

methods like fuzzy c-means clustering are beyond the scope of the thesis and the reader should consult the literature.

### 2.4.2 Optical Flow

Optical flow capturing motion is one of the most dominant low-level cues in the visual system of humans and animals. Hence, reliable estimation of optical flow field is very useful in many computer vision applications from localization, perception of structure or visual grouping. Various methods based on phase correlations, block-based minimization of sum of squared distances or partial derivatives exist, however only a few of them address the problem of large displacements and dense flow field. The two most successful approaches represent recently proposed SIFT-flow [50, 49] and Large Displacement Optical Flow (LDOF) [12].

#### SIFT-flow

SIFT-flow [50, 49] addresses the problem of image alignment at the scene level, which means that it finds the relations between two images of the same scene category, but from different instances.

SIFT-flow employs a state-of-the-art local invariant SIFT descriptor, which is computed for every pixel (dense SIFT). Next, SIFT-flow is formulated as a minimization, very similar to the standard optical flow, however, it is defined over SIFT features instead of RGB values

$$\begin{aligned} E(\mathbf{w}) = &\sum_{\mathbf{x}} \|\mathbf{I}_1(\mathbf{x}) - \mathbf{I}_2(\mathbf{x}+\mathbf{w})\|_{L1} + \frac{1}{\sigma^2}\sum_{\mathbf{x}} \left(u^2(\mathbf{x}) + v^2(\mathbf{x})\right) \\ &+ \sum_{\mathbf{x}_1,\mathbf{x}_2} \min\left(\alpha|u(\mathbf{x}_1) - u(\mathbf{x}_2)|, d\right) + \min\left(\alpha|v(\mathbf{x}_1) - v(\mathbf{x}_2)|, d\right) \end{aligned} \tag{2.2}$$

where $\mathbf{w} = (u(\mathbf{x}), v(\mathbf{x}))^{\mathrm{T}}$ is the flow vector for every pixel $\mathbf{x}$.

An important difference between standard optical flow estimation and SIFT-flow is the size of search window for SIFT flow, which is much larger than that for optical flow since an object can move significant from one image to another in scene alignment.

Since SIFT-flow is much more motivated by alignment of images from different instances, than some video sequence, it is worth mention, that SIFT-flow is a useful method, if the sampling is dense in world images rather than in time.

#### Large Displacement Optical Flow

The above mentioned approach was motivated by image alignment at the scene level (two images from a similar scene category, but from different instances). In contrast, Large Displacement Optical Flow (LDOF) is directly motivated by reliable estimation of an optical flow from video sequences. LDOF [12] builds on the seminal work by Horn and Schunck, who introduced variational methods, where a local, gradient-based matching of pixel values is combined with a global smoothness assumption for dense optical flow estimation.

LDOF extends such an approach with integration of discrete keypoint matches into the continuous energy formulation that is optimized by a coarse-to-fine scheme to estimate large displacements also for small scale structures. The continuous energy function is minimized over color, gradient, smoothing, descriptor matching and its smoothing (integrates discrete matching into continuous, variational model) terms

$$\begin{aligned} E(\mathbf{w}) = &\int_\Omega \Psi_1(|\mathbf{I}_2(\mathbf{x}+\mathbf{w}(\mathbf{x})) - \mathbf{I}_1(\mathbf{x})|^2) + \gamma\Psi_2(|\nabla\mathbf{I}_2(\mathbf{x}+\mathbf{w}(\mathbf{x})) - \nabla\mathbf{I}_1(\mathbf{x})|^2)\, d\mathbf{x} \\ &+ \beta \int_\Omega \delta(\mathbf{x})\rho(\mathbf{x})\Psi_3(|\mathbf{w}(\mathbf{x}) - \mathbf{w}_1(\mathbf{x})|^2)\, d\mathbf{x} + \int_\Omega \delta(\mathbf{x})|\mathbf{f}_2(\mathbf{x}+\mathbf{w}_1(\mathbf{x})) - \mathbf{f}_1(\mathbf{x})|\, d\mathbf{x} \\ &+ \alpha \int_\Omega \Psi_S(|\nabla u(\mathbf{x})|^2 + |\nabla v(\mathbf{x})|^2)\, d\mathbf{x} \end{aligned} \tag{2.3}$$

where $\mathbf{w} = (u, v)^{\mathrm{T}}$ and $\Psi_*(s^2)$ is a general penalizer function with its derivative $\Psi'_*(s^2) > 0$.

LDOF is up to 5× faster than SIFT-flow and is much more memory efficient. Moreover, GPU-based imlementation [79], which deals with parallel variational solver exists and is up to 80× faster than the original implementation.

### 2.4.3 Features

The color features usually are not sufficient to capture the properties of a local patch. Various methods exist, including local invariant features. In this section, we briefly mention Local Binary Patterns (LBP) and textons, since both are used by our similarity metric.

#### LBP

Local Binary Patterns (LBP) represent an efficient approach to the structural description of textures in terms of speed and memory complexity and are discriminative. Their efficiency come from the fact, that in contrast to SIFT and SURF and

expensive computing of gradient distributions they consist of a set of simple binary tests based on a comparison of intensities between the reference pixel and pixels in its neighborhood sampled in an uniform manner. Since the result is simply true or false, the final descriptor is represented as a bit string consisting of $N$-bits.

It should be noted, that recently proposed binary descriptors with non-uniform sampling such as BRIEF, BRISK or ORB are becoming very efficient in a domain of fast feature matching.

#### Textons

Textons are based on a bank of redundant filters. Various banks of filters can be used, e.g. bank consisting of 2D Gaussian filters, 2D Gaussian derivatives in X and/or Y and 2D Laplacian of Gaussian (LoG) in various scales, orientation, ... Usually, only a few distinct filters characterize the texture properties and all others are noisy variations of them.

This is addressed by textons – filters' responses are clustered by k-means algorithm. The associated filter response vectors are called the appearance vectors [45].

### 2.4.4 FH segmentation

Although there are a bunch of papers about image segmentation in the computer vision literature, none of them works perfectly, because such low-level methods are not able to benefit from high-level information about context, scene geometry, ... These are the good reasons why it useful to use an algorithm that will produce over-segmented image regions, so that information about object boundaries, etc. would be preserved and fast.

The algorithm proposed by Felzenszwalb and Huttenlocher [20] for the segmentation of an image into regions is a popular choice since it has (1) ability to preserve detail in low-variability image regions while ignoring detail in high-variability regions, and (2) it has nearly linear time complexity with the number of graph edges.

An image is represented by an undirected graph $G = (V, E)$, where vertices $v_i \in V$ correspond to a pixel in the image, and the edges in $E$ connect certain pairs of vertices $v_i$ and $v_j$. Each edge $(v_i, v_j) \in E$ has a corresponding weight $w(v_i, v_j)$, which is a non-negative measure of the dissimilarity between neighboring elements $v_i$ and $v_j$, e.g. the difference in intensity, color, motion, ...

Edges are then ordered by their weight in a non-decreasing order. The ordered list of edges is traversed one by one, deciding if two pixel regions $C_1$ and $C_2$ connected by the edge considered are merged according to a score measuring the difference between $C_1$ and $C_2$, relative to the internal similarity within $C_1$ and $C_2$. Both the region difference and internal similarity are computed from the existing edges in the graph and no additional measurements in the image are necessary. The algorithm is closely related to Kruskal's algorithm for constructing the minimum spanning tree of a graph, so it can be implemented to run in $\mathcal{O}(m \log m)$, where $m$ is the number of edges in the graph using a disjoint-set forest with union by rank and path compression.

# 3 MONOCULAR CAMERA BASED NAVIGATION OF UGV

*In this chapter, we propose a novel approach – a fusion of the frequency based vanishing point estimation and probabilistically based color segmentation (sec. 3.1). Detection of a vanishing point is based on the estimation of a texture flow, produced by a bank of Gabor wavelets and a voting function (sec. 3.2). Next, the vanishing point defines the training area which is used for self-supervised learning of color models (sec. 3.3). Finally, road patches are selected by measurement of the roadness score. A few rules deal with dark cast shadows, overexposed highlights and adaptivity speed. In addition to the robustness of our system, it is easy-to-use since no time consuming calibration is needed.*

## 3.1 Vision System Design

This section introduces a novel approach to robust detection of shady and highlighted roads by a monocular camera. By comparison with recently presented state-of-the-art methods, [17, 18], we neither use a laser range finder nor stereo vision for extraction of the training area. Our system is based on vanishing point estimation and does not need any time consuming calibration or difficult classifier training or other sensors. Our approach is a fusion of the frequency based estimation of so called **vanishing point** and probabilistically based **texture segmentation**.

Fig. 3.1: **Output of proposed method**: the blue cross is the estimated vanishing point, the yellow trapezoid determines the training area and all non-road regions are overlayed with red (best viewed in color).

A combination of two different approaches allows us to solve difficult situations without any a priori knowledge of a robot's environment. The basic idea of our solution is estimation of the vanishing point, which determines the training area for texture segmentation. Next, road color models are constructed from sample pixels defined by the training area. These models are associated with previously learned models which are stored in a memory. Further, learned models are adaptively updated. Therefore, the models include both the road colors' history and the current road appearance. A few simple rules define properties of the color segmentation system like adaptivity speed, selectivity, robustness or behavior in shady and/or overexposed highlighted road segments.

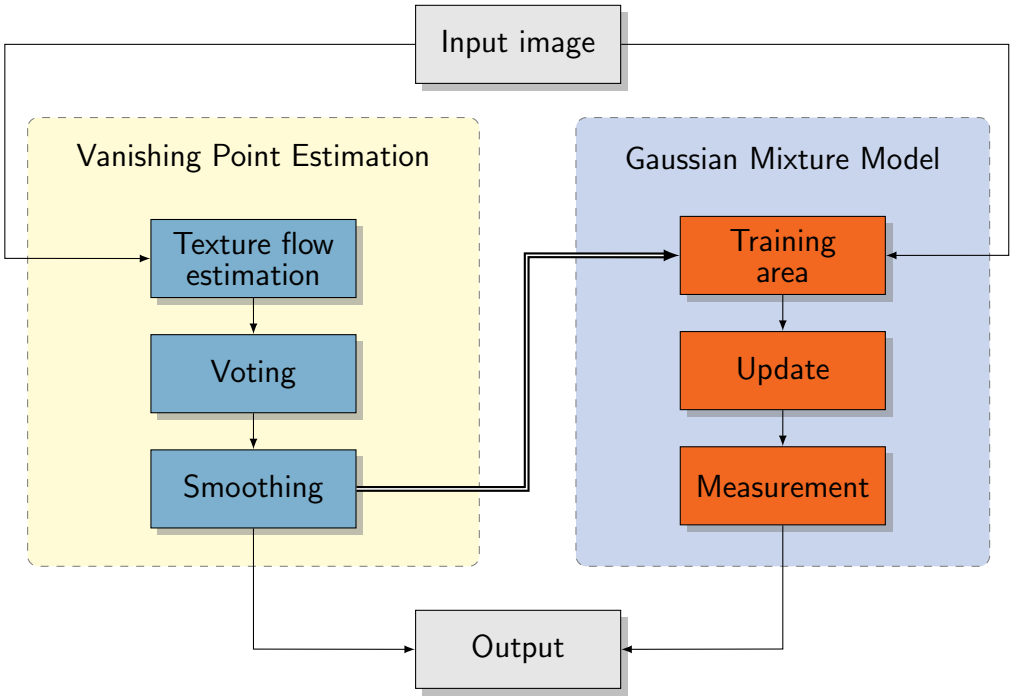
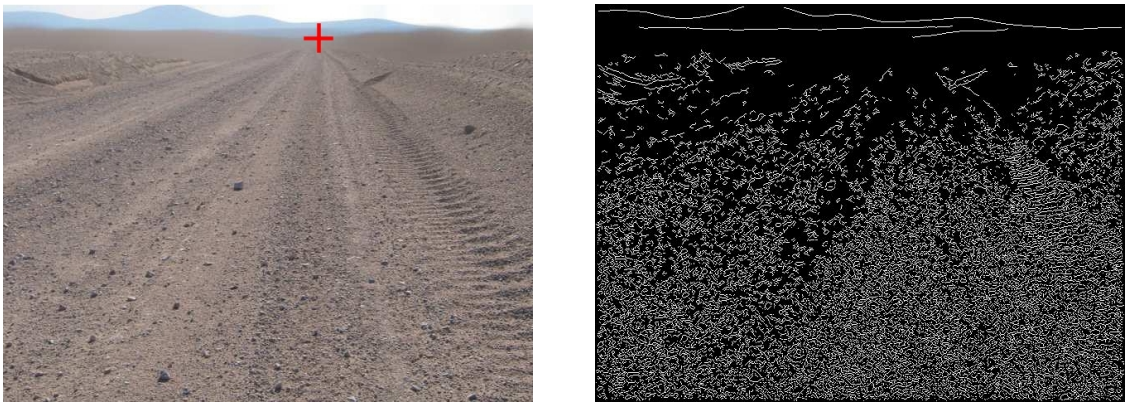



Fig. 3.2: **Overview of proposed method**

The strategy of our vision system is the following: start with the vanishing point estimation which is used to detect the training area for self-supervised learning of color models. Next, self-supervised learning continues, however, it is possible to perform road segmentation based on these models. Besides, a combination of two different approaches is advantageous because in situations like sudden road texture or illumination change, we are still able to estimate the correct course because if the color models are not consistent with current road surface, it is possible to use a vanishing point until new color models are learned.

## 3.2 Vanishing Point Estimation

Parallel lines in the real world do not look like parallel lines under the perspective projection. Therefore, borders of each straight road in an image plane converge at some point, the so called **vanishing point**. For well engineered structured road, it is usually possible to detect this point by a "cascaded" Hough transform, however, such approaches usually completely fail in the case of unstructured roads.



(a) Vanishing point (b) Canny edge detector

Fig. 3.3: **Comparison of a vanishing point and Canny edge detector**, courtesy of [66]

There are a bunch of papers and promising work that utilizes reliable Vanishing Point estimation (see Chapter 2). On the other hand, none of the papers mentioned in the state-of-the-art discuss the crucial issue for mobile robotics: "How to estimate the VP in real-time?", because the original algorithm is quite close to real-time, but not close enough[1]. It is possible to use massive computational power of specialized DSP, FPGA or GPGPU, however, we strongly believe that the main domains of such a guide-path following algorithms are for e.g. primarily teleoperated robots that are able to return in the case of signal loss, a swarm of cheap robots with assisted autonomy, etc. In such cases, it is usually inconvenient (price) or even impossible (weight) to use additional hardware. To our best knowledge, we are the first in who aim at computational efficiency, instead of just a precision of the VP estimation algorithm. In this section we propose a method which achieves results comparable to the methods mentioned in state-of-the-art (Chapter 2), however our method is significantly faster without any dependency on a specific hardware platform.

[1] Rasmussen uses Nvidia GeForce 6800 to accelerates voting.

### 3.2.1 Texture Flow Estimation

The first step of a vanishing point estimation algorithm is, the estimation of a *texture flow* (see Fig. 3.4). The dominant orientation $\theta(\mathbf{p})$ of an image at pixel $\mathbf{p}(x, y)$ describes strongest local parallel structure or texture flow. There exist various techniques, which can be used for estimation of dominant orientation, involving usage of Gaussian pyramids with principle component analysis, steerable filters, etc. We follow the line of research that investigates grouping of dominant orientations, that are estimated by a bank of 2D Gabor wavelet filters, since they are known to be accurate [66, 68, 44]. The kernels of Gabor wavelet filters are quite similar to the 2D receptive field profiles of the mammalian cortical simple cells and show suitable characteristics of spatial locality and orientation selectivity [32].

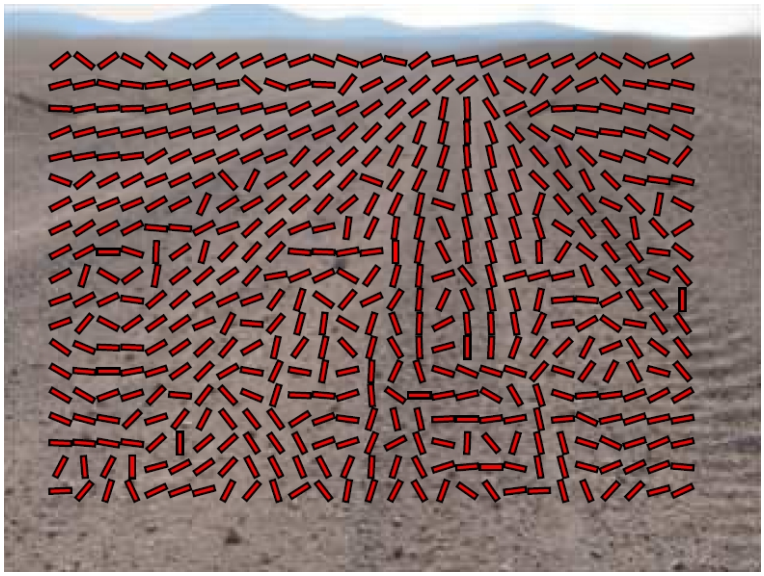
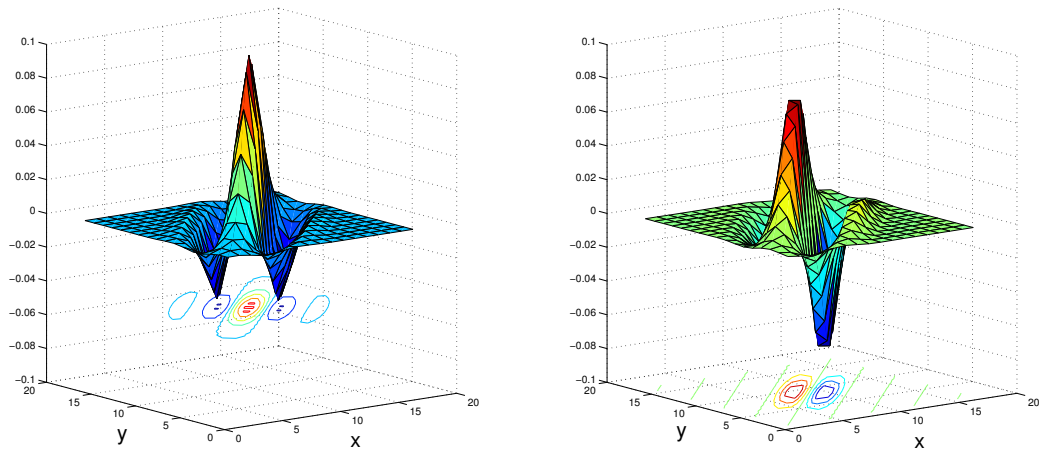


Fig. 3.4: **Texture flow**, courtesy of [66]

#### Gabor wavelet filters

Gabor transformation is a special case of the Short Time Fourier Transform (STFT) which uses windows to determine the frequency and the phase content of the local parts of a signal as it changes over time. It was observed that Gaussian window provides the best trade-off between the product of a time period and bandwidth. Consequently, the 2D Gabor wavelets consist of a product of an elliptical Gaussian and a complex plane wave. The Gabor wavelets are self-similar, which means that all kernels can be constructed from one mother wavelet by its dilation and/or rotation [44].




(a) (b)

Fig. 3.5: **Gabor wavelet**: a) real, b) imaginary part

The set of $k \times k$ Gabor kernels for an orientation $\theta$, radial frequency in radians per unit length $\omega$, scale $s$ and odd or even phase are defined by

$$\psi(x, y, \theta, \omega) = \frac{\omega}{\sqrt{2\pi}c} e^{-\frac{\omega}{8c^2}(4a^2+b^2)} \left( e^{ia\omega} - e^{\frac{c^2}{2}} \right), \tag{3.1}$$

where $x = y = 0$ is the kernel center. Next, $a$ and $b$ are defined by

$$\begin{aligned} a &= x\cos(\theta) + y\sin(\theta), \\ b &= -x\sin(\theta) + y\cos(\theta), \end{aligned} \tag{3.2}$$

and $c = 2.2$, $\omega = \omega_0 \times 2^s$, $\omega_0 = 2.1$ and $s = \{0, 1, 2, 3, 4\}$. More details about 2D Gabor wavelets are given in [44].

Then, $\psi$ 's DC component is subtracted from Gabor kernel, to satisfy one of the design constraints for filters measuring phase disparities to ensure optimal phase

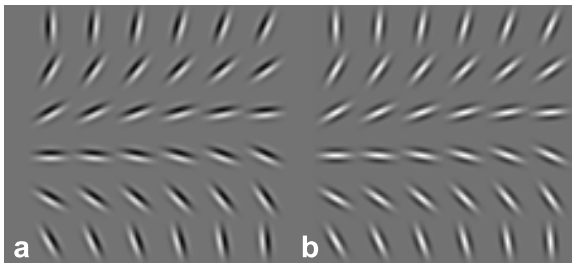



Fig. 3.6: **Single scale Gabor filters**: a) imaginary and (b) real parts of Gabor filters for 36 orientations.

behavior [26]

$$\psi_{DC}(x, y, \theta, \omega) = \psi(x, y, \theta, \omega) - \frac{1}{k^2} \sum_{x=-k/2}^{x=k/2} \sum_{y=-k/2}^{y=k/2} \psi(x, y, \theta, \omega). \tag{3.3}$$

Finally, kernel's coefficients are normalized to make the filter more robust to spurious noise, so that $L_2$ norm is equal to one

$$\psi_{L_2}(x, y, \theta, \omega) = \frac{\psi_{DC}(x, y, \theta, \omega)}{\sqrt{\sum_{x=-k/2}^{x=k/2} \sum_{y=-k/2}^{y=k/2} \psi_{DC}(x, y, \theta, \omega)^2}}. \tag{3.4}$$

Let $i(x, y)$ be the intensity value of a grayscale image at spatial coordinates $(x, y)$. The response of each filter is measured by convolution of an image $i$ with each of $n$ evenly spaced Gabor filter orientations

$$\psi_{L_2}(x, y, \theta, \omega) * i(x, y) = \sum_{m=-k/2}^{m=k/2} \sum_{n=-k/2}^{n=k/2} \psi_{L_2}(x - m, y - n, \theta, \omega) i(m, n) \tag{3.5}$$

where $*$ denotes convolution.

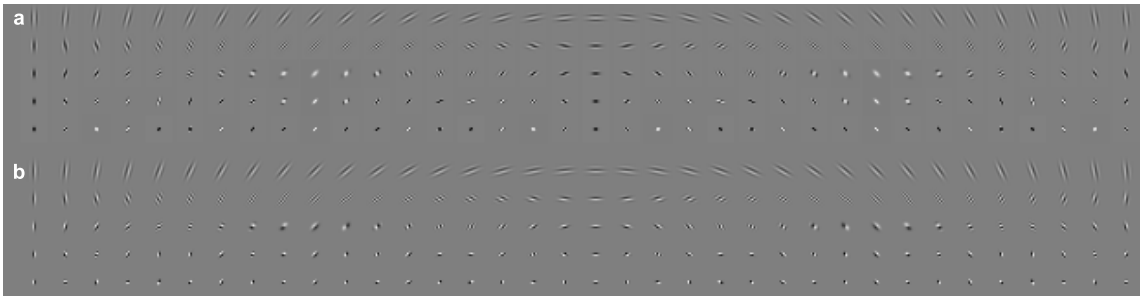



Fig. 3.7: **Multiscale Gabor filters**: a) real and (b) imaginary parts of Gabor filters for 36 orientations and 5 scales.

### Efficient Computation

Computational efficiency is crucial for mobile robots. There exist several ways which deal with this issue. Among them, the most straightforward approach is sub-sampling of an input image and construction of a Gaussian pyramid to compute the responses of multiscale filters more efficiently, since the scale of the filter is constant in this setup. The main drawbacks of such approach are the facts, that we need to be careful with sub-sampling to avoid of creation of artifacts and in addition to that, the filters are still to large for efficient computation of 2D convolution.

Although it is often mentioned, that it is possible to apply convolution theorem and thus the response of the bank of filters might be efficiently computed by FFT

$$\psi_{L_2}(x, y, \theta, \omega) * i(x, y) = \mathcal{F}^{-1}\{\mathcal{F}\{\psi_{L_2}(x, y, \theta, \omega)\}. * \mathcal{F}\{i(x, y)\}\}, \tag{3.6}$$

where $.*$ denotes point-wise multiplication, $\mathcal{F}$ Fourier transform and $\mathcal{F}^{-1}$ inverse Fourier transform, respectively, it was observed, that such computations are not fast enough due to the large number of filters in the bank (consider commonly used 36 orientations and 5 scales) [62]. FFT has approximately logarithmic complexity $\mathcal{O}(N \log_2 N)$, which means that it is more efficient with larger image patches. And finally, the whole image must be processed with FFT, it is not possible to select just a small subset of interesting patches.

Thus, we need to optimize this computational routine by a widely used trick with *integral images*, which is well-known especially in the face recognition[2] domain [88].

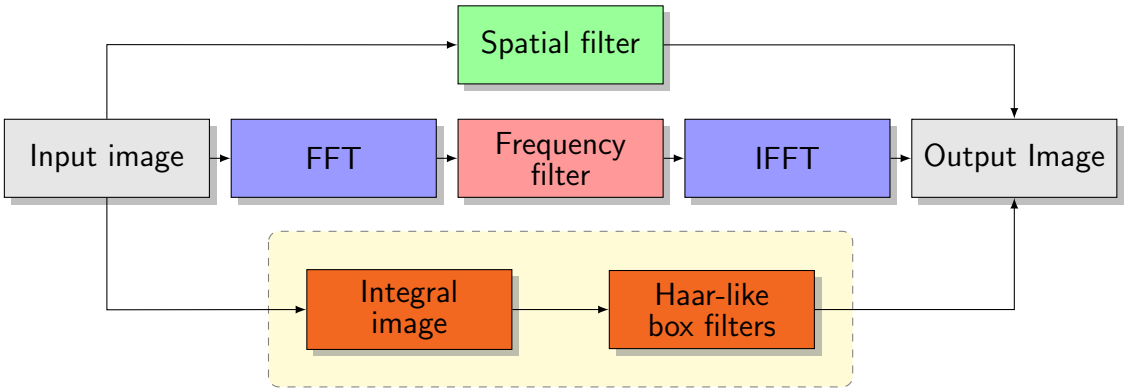



Fig. 3.8: **Approaches to image filtering**

Integral images represent convenient data structure widely used in computer vision, since computational complexity of Haar-like box filters with integral images is independent on the size of the kernel. Integral image can be computed efficiently with a recursive implementation in a single pass over the image as

$$I(x, y) = i(x, y) + A + I(x, y - 1) \tag{3.7}$$

where $I$ is the integral image, $i$ is the input image and $A$ is the accumulate of pixels in the current row. Once the integral image is precomputed, the sum of a rectangular

[2]Integral images were first introduced to the computer graphics comunity as *summed area tables* for texture mapping in 1984 [16], however were not widely used in the computer vision for almost 20 years.

area of any possible size can be computed in a constant time by only three additions

$$\sum_{x'=(left-1)}^{right} \sum_{y'=(top-1)}^{bottom} i(x', y') = I_D + I_A - I_B - I_C, \tag{3.8}$$

where $A$ is top-left, $B$ is top-right, $C$ is bottom-left and $D$ is the bottom-right corner.

It is necessary to decompose the Gabor filters into a linear combination of Haar-like box filters to perform filtration in an integral image domain. First, we define the dictionary $\mathbf{D} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N\}$ of Haar-like box filters, where each $\mathbf{b}_i$ is a column vector formed by reshaping the rectangular box function (defined below). The dictionary consists of single Haar-like box filters which can be formally written as binary functions

$$h_{single}(u, v) = \begin{cases} 1 & u_0 \leq u \leq u_0 + w' - 1 \\ & v_0 \leq v \leq v_0 + h' - 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

where $[u_0, v_0]$ are the coordinates of the top left corner and $w'$ and $h'$ are the size of the white box. Gabor wavelets are (anti)symmetric, therefore we also use a vertically symmetric box function

$$h_{symmetric}(u, v) = \begin{cases} 1 & u_0 \leq u \leq u_0 + w' - 1 \\ & v_0 \leq v \leq v_0 + h' - 1 \\ 1 & w - u_0 - w' + 1 \leq u \leq w - u_0 \\ & h - v_0 - h' + 1 \leq v \leq h - v_0 \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

and horizontally symmetric box function are defined in a similar manner. It is obvious, that dictionary $\mathbf{D}$ of the basis functions (atoms) is over-complete, redundant and non-orthogonal. Next, we need to approximate the Gabor wavelets $\psi$ as a linear combination of atoms $\mathbf{b}$ from dictionary $\mathbf{D}$.

$$\psi \approx \widehat{\psi} = \sum_{i \in \Lambda} c_i \mathbf{b}_i \tag{3.11}$$

It is known, that the problem of finding a global optimum of the approximation is considered to be NP-hard, since the dictionary consists of $H(H+1)W(W+1)/4$ single and $2H(H+1)W(\frac{W}{2}-1)/8$ symmetric atoms for a $W \times H$ large kernels.

The authors of [81, 82] proposed the use of a greedy algorithm called Optimized Orthogonal matching Pursuit (OOMP)[3] [1, 70] that finds a sub-optimal solution and

[3]Terminological remark: *Orthogonal* is related with the full backward orthogonality of the residual in each iteration and the fact, that the reconstruction using OOMP-selected base vectors is orthogonal to the residual.

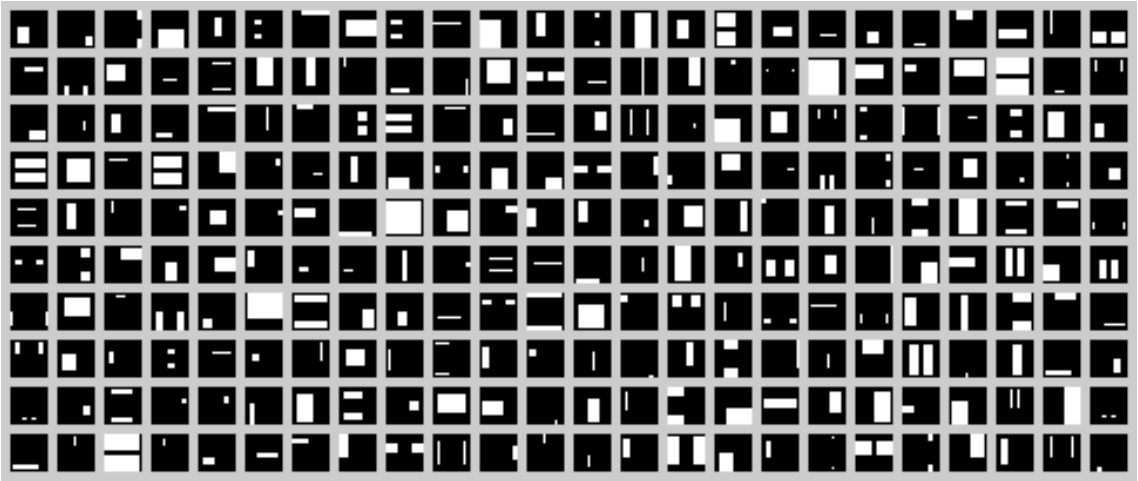


Fig. 3.9: **Dictionary basis**: 240 randomly selected base vectors from the dictionary.

best approximates the original function. OOMP efficiently selects the most representative atoms from an arbitrary redundant nonorthogonal base vector dictionary in a Hilbert space. The OOMP iteratively selects the given number of base vectors $\mathbf{B}_\Lambda = \{\mathbf{b}_{l_1}, \mathbf{b}_{l_1}, \ldots, \mathbf{b}_{l_{|\Lambda|}}\}$ from a dictionary according to the following procedure: suppose that after iteration $\kappa - 1$, the already selected $\kappa - 1$ atoms are defined by the index set $\Lambda_{\kappa-1} = (l_i)_{i=1}^{\kappa-1}$. At iteration $\kappa$, the OOMP selects the index $l_\kappa = i$, that minimizes the new residual, which is equivalent to maximizing

$$\frac{| < \gamma_i, \epsilon_{\kappa-1} > |}{\|\gamma_i\|}, \quad \|\gamma_i\| \neq 0, \quad i \in \overline{\Lambda}_\kappa, \tag{3.12}$$

where $\epsilon_{\kappa-1} = \mathbf{x} - \xi_{\mathbf{B}_{\Lambda_{\kappa-1}}}(\mathbf{x})$ is the reconstruction residual using $\mathbf{B}_{\Lambda_{\kappa-1}}$, $\gamma_i = \mathbf{b}_i - \xi_{\mathbf{B}_{\Lambda_{\kappa-1}}}(\mathbf{b}_i)$ is the component of $\mathbf{b}_i$ that is orthogonal to the subspace spanned by $\mathbf{B}_\Lambda$. $\xi_{\mathbf{B}_{\Lambda_{\kappa-1}}}(\mathbf{x}) = \mathbf{B}_\Lambda(\mathbf{B}_\Lambda^{\mathrm{T}}\mathbf{B}_\Lambda)^{-1}\mathbf{B}_\Lambda^{\mathrm{T}}\mathbf{x}$ is the reconstruction of the signal $\mathbf{x}$ using the nonorthogonal bases indexed by $\Lambda_{\kappa-1}$. $\overline{\Lambda}_{\kappa-1}$ are the indices, that are not selected in the previous $\kappa - 1$ iterations.

As we have already mentioned, it is necessary to remove the DC component of the filter to satisfy one of the design constraints for filters measuring phase disparities and to ensure optimal phase behavior of all filters contained in the bank. This is quite straightforward, if we can easily remove the mean from the filter as is explained in eq. 3.3. The problem is, that the decomposition of the filter into the linear combination consists of coefficients and associated basis functions (Haar-like box filters), that can not be changed (binary functions). Hence, the constraint is obvious: it is necessary to remove the DC component just by tuning the coefficients associated with selected atoms. Let $\chi_{E_i}$ be the characteristic function and $m(E_i)$ is the measure (e.g. square of $L_2$) of the box $\mathbf{b}_i$. Then, the approximated Gabor wavelets with the removed DC


Fig. 3.10: **Decomposition of a Gabor wavelet** into a linear combination of Haar-like box functions.

component are computed as

$$\widehat{\psi}_{DC} = \sum_{i \in \Lambda} \left( c_i - \frac{\delta}{m} \right) \mathbf{b}_i, \tag{3.13}$$

where $\delta = \sum_i c_i m(E_i)$ and $m = \sum_i m(E_i)$. The final kernels $\widehat{\psi}_{L_2}$ are obtained by normalization again, so that $< \widehat{\psi}_{L_2}, \widehat{\psi}_{L_2} >= 1$, i.e. normalized by $L_2$ norm.

Consequently, the standard convolution can be approximated with $N$ Haar-like box filters $\mathbf{b}_i$ selected by OOMP as

$$\varsigma_{\theta,\omega} = \psi_{L_2} * \mathbf{\Omega}_i \approx \widehat{\psi}_{L_2} * \mathbf{\Omega}_I = \sum_{i=1}^{N} \alpha_i (\mathbf{b}_i \mathbf{\Omega}_I), \tag{3.14}$$

where $*$ denotes the convolution operator, $\mathbf{\Omega}$ is the image patch and $\alpha_i$ are the DC corrected and $L_2$ normalized coefficients.

The advantage of computing with an integral image trick is, that the filtering of Gabor wavelets approximated by Haar-like binary box functions with an image patch significantly reduces the number of floating point multiplications and additions. Computing of an integral image (only once per image) requires only $width \times height \times 2$ integer additions with recursive implementation. Each of the $N$ selected atoms consist of one or two boxes. Let $n_{sin}$ be the number of single and $n_{sym}$ be the number of selected symmetric Haar-like box functions (naturally, $N = n_{sin} + n_{sym}$). Then, the approximated convolution needs only $3n_{sin}$ and $7n_{sym}$ integer additions, $N$ floating point multiplications and additions[4]. It is obvious, that the most important parameter, that influence filtering speed is the number of selected atoms $N$.

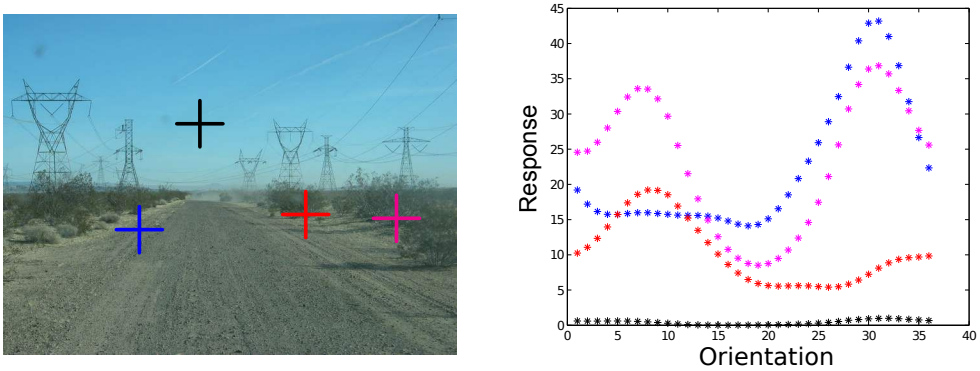
[4]It is expected, that subtraction has the same computational complexity as addition, throughout this chapter.

### Dominant Orientation

Gabor wavelets have two parts: a real and an imaginary component. Thus, an average over the scales of a square norm of the so-called *Gabor energy* (*complex response*) is computed to get the best characteristics of a local texture jet

$$\mathbf{E}_\theta(x,y) = \mathrm{Avg}_\omega \left[\Re(\varsigma_{\theta,\omega}(x,y))^2 + \Im(\varsigma_{\theta,\omega}(x,y))^2\right] \tag{3.15}$$

Rasmussen [66] defines the *dominant orientation of a texture flow* at pixel $\mathbf{p}(x,y)$ as the filter orientation which elicits the maximum complex response at that location, however, it was observed [82, 81], that the estimated dominant orientation is not reliable at all pixels, especially at those, that are not related with road. Kong et al. [39, 38] propose to use a confidence score, which measures how peaky the function $\theta \mapsto \mathbf{E}_{\theta,\omega}(x,y)$ is near the optimum angle $\theta(x,y)$, however their confidence score does not take into account, how many maxima the function have. Thus, we slightly refined the confidence score evaluation, since the peaky function near the optimum angle is not enough - to get the reliable dominant orientation estimation, all point with multiple maxima must be rejected as well.




(a) (b)

Fig. 3.11: **Confidence score**: (a) four points on which the Gabor complex responses are evaluated and (b) the corresponding responses, courtesy [39]

Instead of using of an ordered set of the complex responses, that do not take into account the angle $\theta$ for which the response is measured, we rather pick just the strongest response $e_{max}(x,y)$ from $\mathbf{E}_{\theta,\omega}(x,y) = \{e_1(x,y), e_2(x,y), \ldots, e_A(x,y)\}$ and measure the confidence score as

$$\mathrm{Conf} = 1 - \frac{\mathrm{Avg}\ \vartheta}{e_{max}(x,y)}, \tag{3.16}$$

$$\vartheta = \{e_{max-b}(x,y), \ldots, e_{max}(x,y), \ldots, e_{max+b}(x,y)\}, \tag{3.17}$$

where $A$ is the number of orientations and $b$ is the coefficient, that determines how much weaker the other responses are expected to be (we use $b = \frac{A}{4} - 1$). In addition to that, all other responses from $\overline{\vartheta}$ (complement of the set $\vartheta$) are compared with Avg $\vartheta$ and if any of them is higher than Avg $\vartheta$, the confidence score is set to zero to reject the pixel with multiple maxima. Next, the confidence score is normalized to the range $< 0, 1 >$ and threshold, so that all pixels with a confidence level lower than T = 0.3 are discarded.

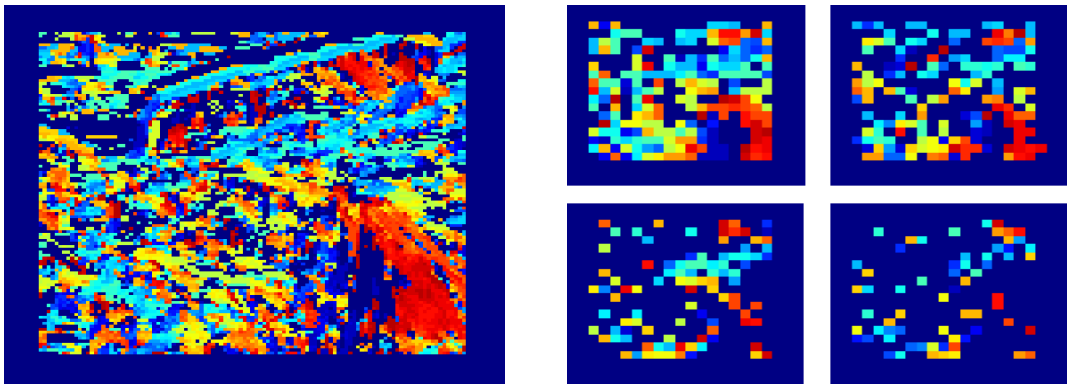
### 3.2.2 Vanishing Point Voting

The second stage of the vanishing point estimation algorithm is voting. The idea behind the voting scheme assumes, that the set of parallel lines in the 3D space do not look like a parallel under the perspective projection caused by a pinhole camera, however these lines converge to some point on the image plane, the so-called vanishing point. The vanishing point is important in many computer vision applications, e.g. structure from motion [2], robotics, ... One can argue, that a single vanishing point have only straight roads and there might be multiple vanishing points in the case of a curved road. This is not a constraint, since we simply estimate the strongest response. The original voting scheme proposed by Rasmussen [66] is the tightest bottleneck of the whole algorithm. Kong et al. [39, 38] proposed a locally adapted soft-voting scheme, that slightly reduces the computational complexity of the original algorithm, however it is still far away from real-time.

Hence, the first step of our algorithm is a reduction of vanishing point candidates (the number of voting pixels were reduced by the confidence score). The idea is that the vanishing point should lie close to the points with significant dominant orientation, since the locally adapted voting strategy allows voting only to the points, that are in a close supporting subregion (discussed below). Thus, we take into account only those points, that are not rejected in the previous stage (confidence score based thresholding) and perform simple morphological dilation to include pixels, that are close to the huge support region, however they do not have significant dominant orientation themselves. The set of these points $C$ are the vanishing point candidates. It is possible to use more sophisticated algorithms, however we use only such a simple preprocessing filter due to the very low computational complexity.

Next, we introduce superpixels into the voting scheme: a histogram of orientations is computed for each $f \times f$ subregion in the image and if the population of the histogram maxima is higher than some threshold $\tau = 0.5(f \times f)$, the angle which corresponds with the maxima is associated with the current superpixel and the next

(a)



(b) (c)

Fig. 3.12: **Vanishing point estimation - superpixels**: a) input image, b) dominant orientations, c) superpixels (4 levels)

$l-1$ strongest orientations in histogram are compared with another threshold, equal to some fraction (0.5) of the histogram maxima frequency. If the frequencies of these additional $l-1$ orientations are higher than this threshold, they are associated with the current superpixel as well. To preserve the character of the superpixel, we not only store orientations, but also their histogram frequencies.

Once we have the set of possible vanishing point candidates $C$, one can make to vote the pixels with an estimated dominant orientation to obtain the vanishing point. Formally, let the angle of the line joining an image pixel $\mathbf{p}$ and a vanishing point candidate $\mathbf{v}$ is $\alpha(\mathbf{p}, \mathbf{v})$, then $\mathbf{p}$ votes for $\mathbf{v}$ if the difference between $\alpha(\mathbf{p}, \mathbf{v})$ and $\theta_{max}(\mathbf{p})$ is within the dominant orientation estimator's angular resolution, which has a finite value of $\frac{\pi}{n}$. This hard-voting strategy proposed by Rasmussen [66] has one drawback – voting pixels are all pixels below the vanishing point candidate and voting pixels are not weighted by the distance to the vanishing point candidate. This caused, that the vanishing point candidates higher in the image to have more

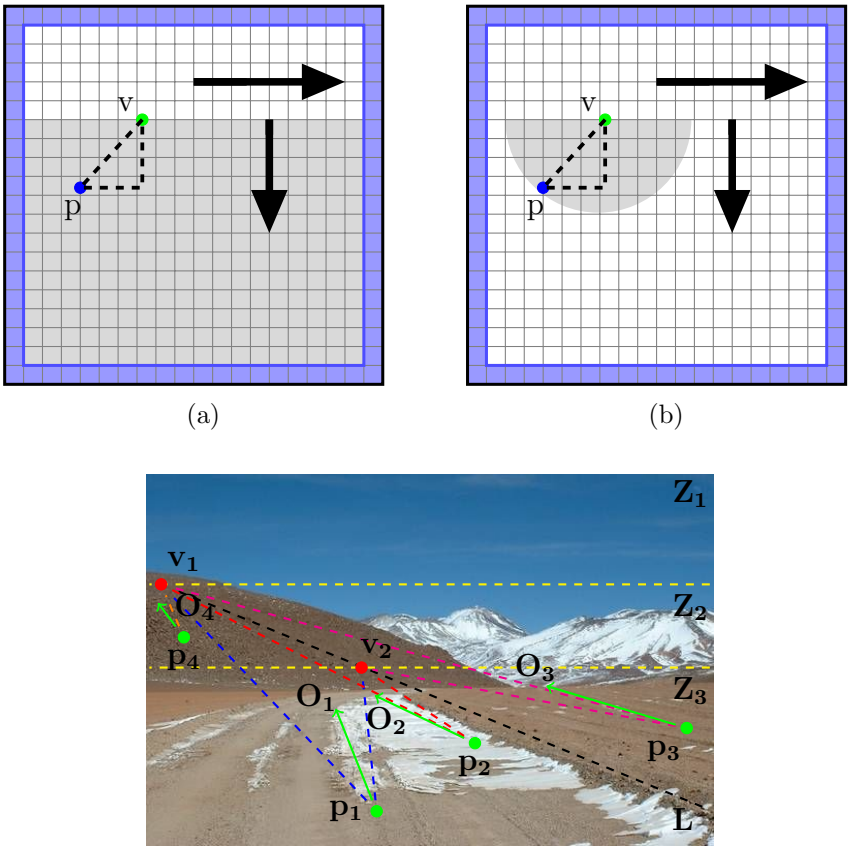



Fig. 3.13: **Voting strategies**: (a) global, (b) local – pixels outside of the blue rectangle are excluded from voting owing to the Gabor kernel size, (c) Illustration of the problem in vanishing point estimation by global hard-voting strategy. $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$ are four possible voters. $\mathbf{v}_1$ and $\mathbf{v}_2$ are two vanishing point candidates (assuming that $\mathbf{v}_2$ is the true vanishing point). $O_1$, $O_2$, $O_3$ and $O_4$ are respectively the texture orientation vectors of the four voters. The two vanishing point candidates divide the whole image region into three zones, denoted as $Z_1$, $Z_2$ and $Z_3$. $Z_1$ does not vote for both candidates. Both $Z_2$ and $Z_3$ potentially vote for $\mathbf{v}_1$ while $\mathbf{v}_2$ receives votes only from $Z_3$. Therefore, the higher vanishing point candidates tend to receive more votes than the lower candidates., courtesy of [39]

potential voting pixels and it may lead to false detection. To overcome this limitation, Kong et al. [39, 38] introduced locally adaptive soft-voting, that reduce the region of voting pixels $R$ to an intersection of the Gabor response image with a half-disk below the vanishing point candidate $\mathbf{v}$ centered at $\mathbf{v}$. A radius of the half-disk is $r = 0.35\Gamma$, where $\Gamma$ is the length of the image diagonal. Each pixel with significant dominant orientation from $R$ can vote for vanishing point candidate $\mathbf{v}$ if the following

condition is satisfied

$$vote(\mathbf{p},\mathbf{v}) = \begin{cases} \frac{1}{1+[\gamma\, d(\mathbf{p},\mathbf{v})]^2} & \text{if} \quad \gamma \leq \frac{5}{1+2d(\mathbf{p},\mathbf{v})}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.18}$$

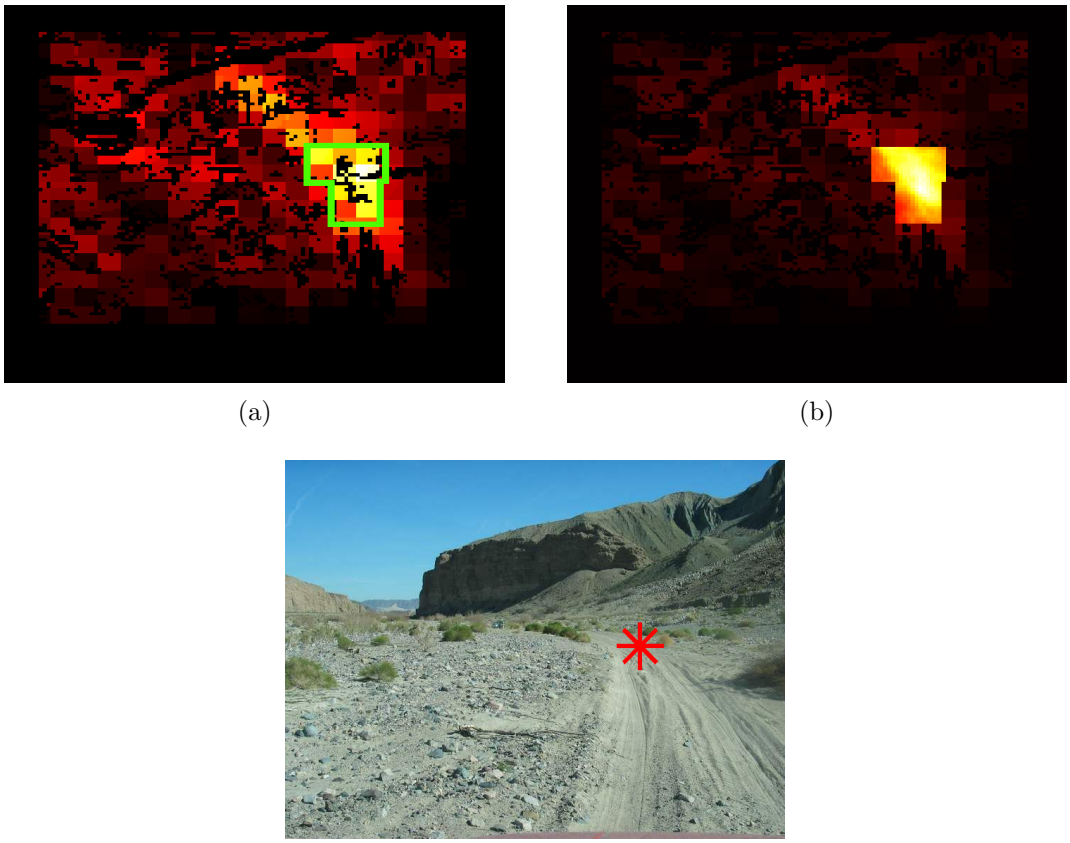
where $\gamma = |\alpha(\mathbf{p},\mathbf{v}) - \theta_{max}(\mathbf{p})|$.

Next, the definition of an objective function for each vanishing point candidate $\mathbf{v}$ is straightforward

$$votes(\mathbf{v}) = \sum_{\mathbf{p}\in R(\mathbf{v})} vote(\mathbf{p},\mathbf{v}). \tag{3.19}$$

Here we discuss how we reduce the computational complexity of voting. Our voting consists of two stages. First, we vote for all possible vanishing point candidates from $C$, however only sparse superpixel representation is used for voting. Thus, we get a coarse estimation where the vanishing point approximately lies. Next, we find the maxima of the objective function and establish a subregion (rectangle of a size $j$) around this point. Then, we check, if there are some points with a higher score than some fraction (0.8) of the global maxima. If such points exist, we establish another subregion around this point and iteratively re-scan all other pixels again, until there are no more pixels with a score higher than the fraction of the global maxima. Finally, the union of such regions (usually, there are only $1-3$) is used in the second stage of voting – the score of these pixels is cleared and voting is performed again, however, original dominant orientations are used instead of superpixels. Hence, we are able to estimate the vanishing point with the same precision and a huge reduction in computational complexity.

The differences in a computational complexity of various voting schemes are significant. Both, Rasmussen [66] and Kong et al. [39, 38] as well, consider all image points as vanishing point candidates. The latter approach is faster, since not all points below the vanishing point candidate can vote - the voting region is reduced to a half-disk with radius $r$, however, all pixels of $w \times h$ image are still considered as vanishing point candidates. In contrast to preceding voting schemes, our approach reduces the number of vanishing point candidates, since we consider only points with a high confidence score. Moreover, by utilization of integral image filtering, we do not need to estimate the dominant orientation at all pixels as in the case of FFT, however, we can easily discard all pixels with low variance. This variance filter can also be efficiently computed in an integral image domain. Based on our observations, usually approximately 50% of pixels are rejected. In the second step, the size of the voting region is reduced by factor $f$, since we use superpixels. Thus, we get a rough estimation of vanishing point coordinates in a very cheap way and only the union of



(a) (b)

(c)

Fig. 3.14: **Voting – coarse-to-fine**: (a) coarse estimation and an area with highest score (green), (b) refinement, (c) output.

subregions with score close to the maxima are re-voted. Usually, there are only $1-3$ subregions used for refinement, which have the same computational complexity as Kong et al. [39, 38], however only a few pixels are considered as candidates (consider $1-3$ subregions of size $8 \times 8$ against the full region with size of $128 \times 128$).

### 3.2.3 Temporal Smoothing

One can see, that extraction of a vanishing point from objective function is straightforward - it is pixel, where the number of votes elicits its maximum. Instead of usage of output independently per each frame, we rather run a smoothing filter throughout the whole sequence to reduce influence of noise and to avoid the jumpy characteristic of output. Particle filters (sequential Monte Carlo) are often used in computer vision since they overcome many limiting assumptions of Kalman Filters.

State of the system $X_t = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t\}$ at time $t$ have a probability density

function telling us what $\mathbf{x}_t$ likely to be. This is represented by a set of $N$ particles (sample states). We also have a set of observations $Z_t = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t\}$ which are probabilistically related to the state $\{\mathbf{x}_i\}$ and correspond to the objective function. Due to the Markovian assumption, that $\{\mathbf{x}_t\}$ depends probabilistically on the preceding state $\{\mathbf{x}_{t-1}\}$, we can model $P(\mathbf{x}_t|\mathbf{x}_{t-1})$.

Particle filters are successful in the tracking of multimodal distributions. Unfortunately, objective function in an urban environment usually do not have sharp a maximum which is necessary for correct prediction. Thus, a DC component is subtracted from objective function

$$V_{DC}(x,y) = V(x,y) - \frac{\sum\sum_{a,b\in V} V(x,y)}{(I_w - k)(I_h - k)}, \tag{3.20}$$

where V(x,y) denotes voting function. Negative values which are introduced by this subtraction are removed

$$V_{DCcorr}(x,y) = \begin{cases} V_{DC}(x,y) & \text{if} \quad V_{DC}(x,y) > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{3.21}$$

After this preprocessing, a standard CONDENSATION algorithm proposed by Isard and Blake [31] is performed and the best estimate of the state of the system $\mathbf{x}_t$ in time $t$ is given by

$$\mathbf{x}_t = \sum_{i=1}^{i=N} \pi_{t,i}\mathbf{s}_{t,i}, \tag{3.22}$$

where $\mathbf{s}_{t,i}$ represents samples (particles) in time $t$, $N$ is a number of particles and $\pi_{t,i}$ are their associated weights.

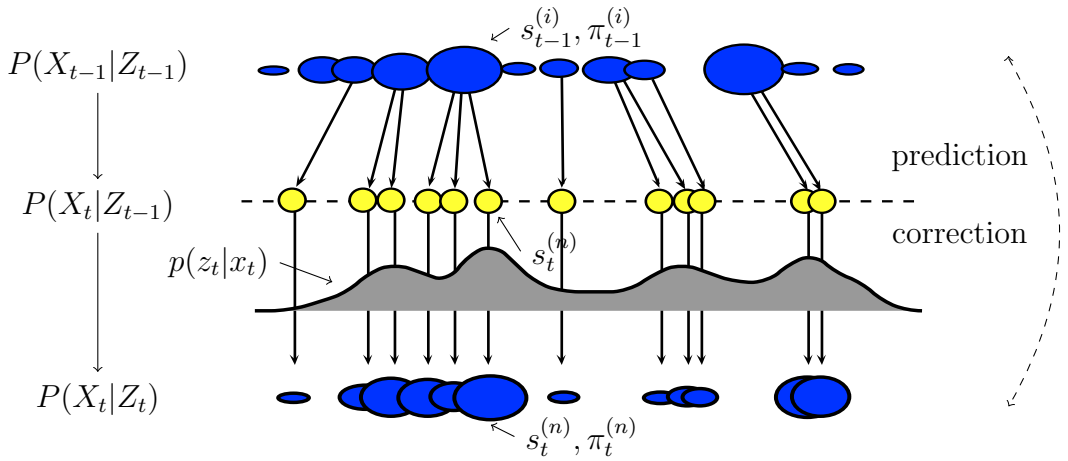
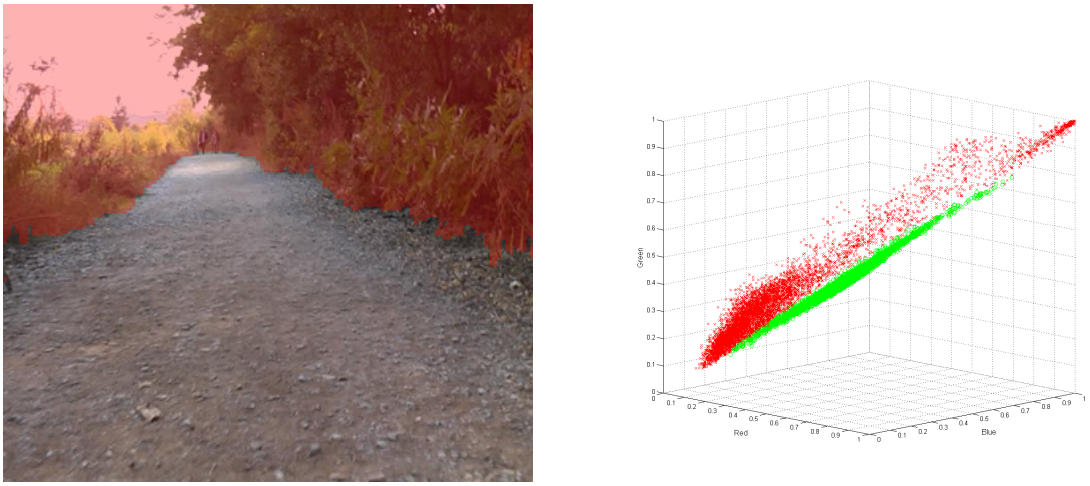



Fig. 3.15: **CONDENSATION**: CONditional DENSity propagATION.

## 3.3 Road Extraction

In fact, a vanishing point does not tell us anything about a road surface. Vanishing points provide information about direction, however we do not have any information about free space ahead of the robot. Thus, another algorithm based on adaptive color segmentation is needed. We chose algorithm based on Gaussian Mixture Models (GMM) and self-supervised learning (see Fig. 5.10).

There are several reasons why Gaussian Mixture Models are useful. The main idea behind this concept lies in color representation. Although RGB space forms a well-known cube, most colors at natural scenes form some lines which begin in a very close distance from the black color (c.f. Fig. 3.16). In fact, this is the reason why naive approaches such as thresholding that uses the cube ($L_{max}$) or sphere ($L_2$) distances fail in a natural environment. The more discriminative metric is necessary.




(a) (b)

Fig. 3.16: **Colors in a natural scene**: (a) input image and (b) pixel distribution in RGB cube. Green points correspond to the road area, while the red points denote the overlayed non-road region.

GMMs represent convenient structure, since each model is fully determined by its mean vector $\boldsymbol{\mu}$, covariance matrix $\boldsymbol{\Sigma}$ and a number of associated pixels. The position of each Gaussian in a space is determined by its mean vector; eigenvalues and eigenvectors describe the shape and rotation, respectively. Mahalanobis distance, which can be viewed as a straightforward generalization of Euclidean distance is used to measure similarity between the model and an input pixel.

### 3.3.1 Training Area

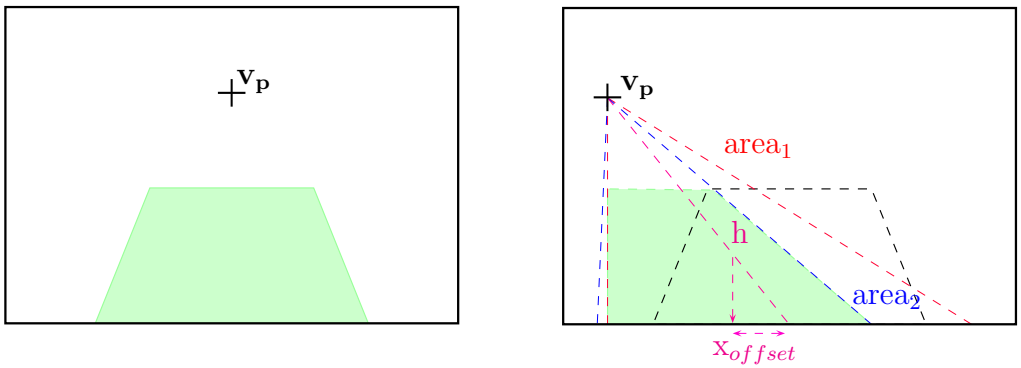
By comparison with previously published algorithms [17, 18], the training area is determined by the estimated vanishing point. The training area is initialized in its default position – centered trapezoid at the bottom of the image. Next, to remove non-road regions, the training area is shifted

$$x_{offset} = \frac{(I_h - h)(v_x - I_{\frac{w}{2}})}{I_h - v_y}, \tag{3.23}$$

where $I_h$ is image height, $I_{\frac{w}{2}}$ is half of image width, $v_x$ and $v_y$ are spatial coordinates of vanishing point and $h$ denotes projection constant, which is set to the half of height of a training area.

After transition of the default training area to the new position, two regions are settled. The first one ($area_1$) is delimited by lines joining the vanishing point $\mathbf{v}_p$ and ending points of the polygon's base, the second one ($area_2$) is created by lines joining the vanishing point with bumpers (approx. 10 pixels from image boundaries). The final shape of the training area is computed as an intersection of $area_1$ and $area_2$ (c.f. Fig. 5.9).

$$area = area_1 \cap area_2. \tag{3.24}$$




(a) (b)

Fig. 3.17: **Training area** in (a) its default position and (b) shifted training area.

### 3.3.2 Color Models Management

Next, we describe handling with color models, which are learned from samples defined by the training area. GMM based segmentation can be performed in an arbitrary color space. The experiments showed, that segmentation based on RGB color space works well, however, if environment allows us to use less selective color space, we strongly recommend brightness-invariant $c_1c_2c_3$ color space [22] which successfully deals with uncertain illumination.

$$c_1c_2c_3 = \begin{cases} c_1 & = \arctan \frac{r}{\max(g,b)}, \\ c_2 & = \arctan \frac{g}{\max(r,b)}, \\ c_3 & = \arctan \frac{b}{\max(r,g)}. \end{cases} \tag{3.25}$$

**Construction**

Once the training area is defined, the next step is the building of the Gaussian mixture models (GMM), which are used to detect the road outside of the training area. Instead of a commonly used expectation-maximization (EM), we would rather use a hierarchical agglomerative (bottom-up) k-means clustering (HAC). K-means is a greedy algorithm which iteratively performs the following two operations: (1) assign data vectors to the nearest of the $k$ centers; (2) update each of the $k$ centers to the mean of the data vectors assigned to it. These two steps are continued until the algorithm converges and a local minimum of the criterion is achieved. K-means represents good trade-off between computational complexity and accuracy (covariance matrices are almost the same as with EM). The biggest advantage of HAC is, that the number of models $c$ are not fixed to some value, but is adaptable with the different types of road surface (clusters are merged in the same way, as is discussed in the subsection Update). Each cluster $c$ is represented by its mean vector $\boldsymbol{\mu}$, covariance matrix $\boldsymbol{\Sigma}$ and a mass (the number of pixels associated to each cluster)

$$\boldsymbol{\mu}_c = \frac{1}{n_c} \sum_{i=1}^{i=n_c} \mathbf{p}_{c,i}, \tag{3.26}$$

$$\boldsymbol{\Sigma}_c = \frac{1}{n_c} \sum_{i=1}^{i=n_c} \mathbf{p}_{c,i}\mathbf{p}_{c,i}^{\mathrm{T}} - \boldsymbol{\mu}_c\boldsymbol{\mu}_c^{\mathrm{T}}, \tag{3.27}$$

$$mass_c = n_c \tag{3.28}$$

To ensure robustness of training models, all models which do not have at least $T_{outliers} = 15\%$ pixels of the most massive model are refused as outliers. To avoid troubles with uniformly colored roads, an explicit minimum noise term $\phi I_{3x3}$ is added to the covariance matrix. Another possibility of creating models, is usage of a fuzzy c-means clustering, which allows to be more/less selective whether the point belongs to the cluster or not, instead of standard k-means.

### Update

In addition to $c$ training models, $n_l$ learned models exist, which represent "history of the road" with exponential forgetting. At the beginning, all color models are null. Each training model is compared with learned models

$$(\boldsymbol{\mu}_L - \boldsymbol{\mu}_T)^{\mathrm{T}}(\boldsymbol{\Sigma}_L + \boldsymbol{\Sigma}_T)^{-1}(\boldsymbol{\mu}_L - \boldsymbol{\mu}_T) \leq d_{similar}, \tag{3.29}$$

where $\boldsymbol{\mu}$ is a mean vector, and $\boldsymbol{\Sigma}$ is a covariance matrix. If the training model overlaps any learned model, the learned model is updated according to formulas

$$\boldsymbol{\mu}_{updated} = \frac{m_L\boldsymbol{\mu}_L + m_T\boldsymbol{\mu}_T}{m_L + m_T}, \tag{3.30}$$

$$\boldsymbol{\Sigma}_{updated} = \frac{m_L\boldsymbol{\Sigma}_L + m_T\boldsymbol{\Sigma}_T}{m_L + m_T}, \tag{3.31}$$

$$m_{updated} = m_L + m_T, \tag{3.32}$$

where $m$ is associated mass to the model. Otherwise, there are two possibilities. If all models are not full, then the new model is created. If all models are full, then the model with the lowest mass is discarded and a new one is created in its place.

### Shadows and overexposed highlights

Once the robot is among a shady and/or overexposed highlighted road segments, models with the same original color could be easily discarded after a few frames. It is the same situation when the robot moves away from these parts, however, more shady and/or highlighted road segments are straight forward. Thus, the models with high mass are compared with those with low mass. If the mean color of those models are similar, the mass of small models is adjusted to above some value ($f_{shadow}$ multiplied by the mass of the most massive model). The comparison of mean colors is based on modified Hue proposed by Finlayson [21]. The models are similar, if both conditions are satisfied

$$|Hue(\boldsymbol{\mu}_i) - Hue(\boldsymbol{\mu}_j)| < H_T, \tag{3.33}$$

$$|Brightness(\boldsymbol{\mu}_i) - Brightness(\boldsymbol{\mu}_j)| > B_T, \tag{3.34}$$

$$Hue(\boldsymbol{\mu}_i) = \arctan\frac{\log r_i - \log g_i}{\log r_i + \log g_i - 2\log b_i}, \tag{3.35}$$

$$Brightness(\boldsymbol{\mu}_i) = \frac{r_i + b_i + g_i}{3}. \tag{3.36}$$

In addition to that, shadow and highlight "preprocessors" provides more information about the environment to higher AI [4]. It is important in situations when a robot is not yet among the shadowed/highlighted segments, however, these difficult illumination conditions are straight forward. Without preprocessors, a huge dark

shadows, or overexposed highlights will be labeled as a non-road. Only pixels under line **l** determined by the vanishing point are considered. Both detectors are similar - intensity of each pixel is compared with some threshold and if the value is close enough to 0 for shadow or 1 for highlight preprocesor, pixel is masked

$$shadows(x,y) = \begin{cases} 1 & \text{if } intensity(\mathbf{p}) < T_{shadow}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.37}$$

$$highlights(x,y) = \begin{cases} 1 & \text{if } intensity(\mathbf{p}) > T_{highlight}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.38}$$

where $intensity(\mathbf{p}) = 0.299r + 0.587g + 0.114b$. Unfortunately, masked pixels do not contain enough information about color, thus, these pixels are not automatically labeled as road, however information about these regions are important for higher AI.

### 3.3.3 Adaptivity and Robustness

The mass of each model is an important value for road segmentation (discussed below). However, mass updating formula has an integral character. Consequently, this increases the robustness of the method, however it negatively influences speed of adaptivity. It is possible to solve this naively by a huge decay factor, which is taken off from mass at each frame, however this solution leads to the loss of models history (models do not remember more than last few frames). It is a similar task to the problem of anti-windup, which is well known from control theory of feedback systems. Good choice of appropriate limit is important, because it depends on the number of expected clusters produced by k-means (it expects the worst case - uniformly associated pixels to each cluster), adaptivity speed, which describes the worst case of how many frames it will take before the new model is used, and a factor $d_{classify}$ which is the worst case of threshold used for Mahalanobis scoring (discussed below). Thus, we add saturation nonlinearity with superior limit

$$AWU = \frac{n_{frames_1}}{d_{classify}} \frac{n_{tr}}{c}, \tag{3.39}$$

where $n_{frames_1}$ is a number of frames which determines adaptivity speed, $n_{tr}$ is size of a centered training area, $c$ is an expected number of training models produced by HAC and $d_{classify}$ is a threshold for Mahalanobis distance measurement. Therefore we are able to set adaptivity speed without loss of models history (c.f. Fig. 5.8).

On the other hand, we do not want to store models which were not updated for many frames. Hence, a decay factor from each learned model is taken off in each frame

$$D = AWU^{-\frac{1}{n_{frames_2}}}, \tag{3.40}$$

where $n_{frames_2}$ is a number of frames for exponential forgetting.

### 3.3.4 Road Segmentation

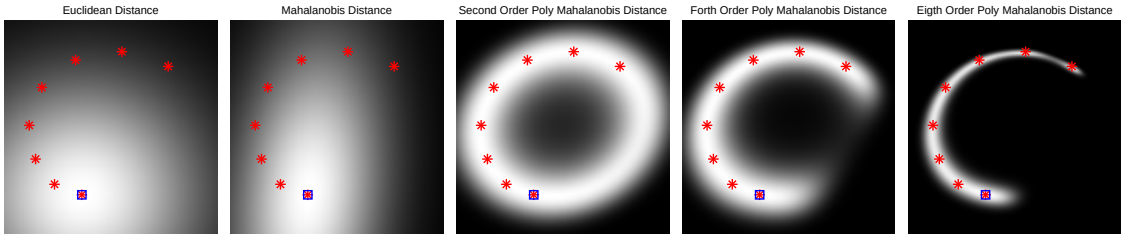
Once all routines connected with management of models are done, we are able to measure a degree of belonging to the road/non-road region of pixels outside the training area. All pixels of the image are assigned a "roadness" score, which is measured as a minimum of the Mahalanobis distance between each pixel and learned models. Only models with mass above some value $d_{classify}$ (fraction of the biggest model) are considered. The condition is important for both reasons - it improves the robustness of the method and saves computation time. The roadness score is measured as a minimum of Mahalanobis square norm

$$D(\mathbf{p}, \boldsymbol{\mu}_i) = \min_i((\mathbf{p} - \boldsymbol{\mu}_i)^{\mathrm{T}} \boldsymbol{\Sigma}_i^{-1} (\mathbf{p} - \boldsymbol{\mu}_i)) \tag{3.41}$$

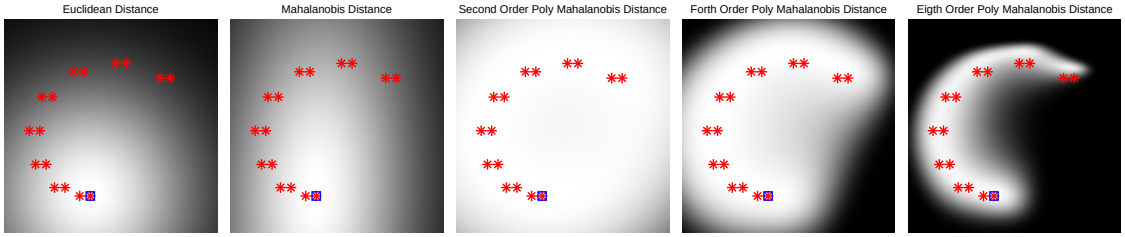
Next, it is possible to use these values as an input of probabilities to some higher AI (e.g. occupancy grids, ...), or identify patches that create the road. To extract only road segments, we run thresholding with an adaptive threshold. The default threshold is determined by pixels belonging to the training area (pixels labeled as outliers by k-means are excluded) – the threshold is set to $\mu+3\beta\sigma$, which ensure that all pixels in the training area are selected as road pixels. Nevertheless, we expect that at least 25% of image is created by non-road pixels. Thus, once the thresholding is done, the non-road pixels are summed up. If the number of non-road pixels is below 25%, parameter $\beta$ decreases and thresholding start again. In fact, it usually is $\beta = 1$, however if the $c_1c_2c_3$ color model fails, $\beta$ decreases to ensure correct classification, but these situations are rare. To remove small areas labeled as non-road and preserve large obstacles, morphological operations dilation an erosion are performed. Finally, only that blob, which is connected with the training area by flood fill, is preserved as a road region, others are discarded as non-road.

### 3.3.5 A Note on Polynomial Mahalanobis Distance

Many algorithms employ Mahalanobis distance, since Mahalanobis distance better follows the data distribution, however it is assumed, that data points have a normal distribution. Recently proposed Polynomial Mahalanobis Distance represents more discriminative metric [23], which provides superior results in an unstructured terrain, especially, if the road is barely visible even for humans.




(a) Simple nonlinear distribution of sample points




(b) Duplicated nonlinear distribution of sample points

Fig. 3.18: **PMD – synthetic example**: synthetic sample points distribution (the brighter the background is, the closer to the reference point denoted by the blue rectangle).

Figure 3.18 compares Euclidean distance, standard Mahalanobis distance and $q$-order PMD [5]. In conformity with theoretical assumptions, Euclidean distance does not follow the distribution of sample points and uses equal scales in all directions (sphere). By utilizing of a covariance matrix, standard Mahalanobis distance shrinks Euclidean sphere into ellipsoid, because it weights the distance calculation according to the statistical variation of each component. As it is shown in Figure 3.18 a), neither Euclidean, nor standard Mahalanobis distance respect highly non-linear character of sample points. Thus, the superior results are obtained with PMD, because it maps its feature points into higher order polynomial terms.

Unfortunately, the situation is a little bit more complicated - it is impossible to say, that a $q$-order PMD outperforms other metrics in all cases. The problem is,

[5] Code provided by Grudic and Mulligen [23] is used for all computations of PMD.

that PMD expects, that the sample points could be easily approximated by a higher order polynomial term. As it is shown in Figure 3.18 b), even with only a small number of similar points (and of course, the situation is even worse in the case of outliers), PMD degenerates and do not follow the sample points precisely. One can demur, that only a $8^{th}$-order PMD is used and that the higher order PMD will follow the sample points much more tightly. It is truth, however it is necessary to consider the fact, that only a few points are used to create the PMD model in this synthetic example, while training areas in real application consist from thousands of pixels[6].

All above mentioned aspects of properties of a PMD may be useful for some kind of a user-guided segmentation, however are quite inconvenient for autonomous systems. In the case of autonomous systems, no specific size of a training area is required - the results should be stable over various sizes. Similarly, we prefer stability over various orders of PMD and $\sigma^2$, since otherwise, we are not able to decide, which order of the PMD and a value of $\sigma^2$ is enough to ensure correct segmentation of a current frame. Moreover, computational complexity is growing with the increasing order of PMD and stability is decreasing with lower values of $\sigma^2$, hence we would like to use lower orders of the PMD with higher values of $\sigma^2$.

(a) Input images

(b) Polynomial Mahalanobis Distance with the three stage algorithm

Fig. 3.19: **PMD – results**.

Although we proposed a model [71] which deals stability of PMD and obtained a very first results (Fig. 3.19), the design of a robust model is still an opened question. Details are beyond the scope of this thesis, reader should consult [71, 23].

[6]Consider the training area of size, e.g. $100 \times 200 = 20000$ pixels.

# 4 SPATIO-TEMPORAL CONSISTENCY

*This chapter addresses the problem of spatio-temporal consistency for dynamic scene labeling. First, we discuss total scene understanding in section 4.1, next we show the advantages and drawbacks of hierarchical inference machines in section 4.2 and give an overview of our smoothing filter in section 4.3. Section 4.4 deals with optical flow estimation and section 4.5 introduces a novel, similarity metric. We put everything together in section 4.6. Final comments are given in section 4.7. Several naive approaches are demonstrated in Appendix A.*

## 4.1 Total Scene Understanding

Everything that we have discussed so far was closely connected with visual navigation of a small semi-autonomous robot (e.g. Orpheus-AC). Usually, such robots are teleoperated for the biggest portion of time, however situations exist (e.g. signal loss, ...) when the robot must operate autonomously. Hence, the most important demand on such system is its universality, so that it can be used in every possible environment as fast as possible without any difficult calibration, training or setup. It is obvious, that a self-supervised learning algorithms (e.g. proposed in chapter 3) are likely the best option.

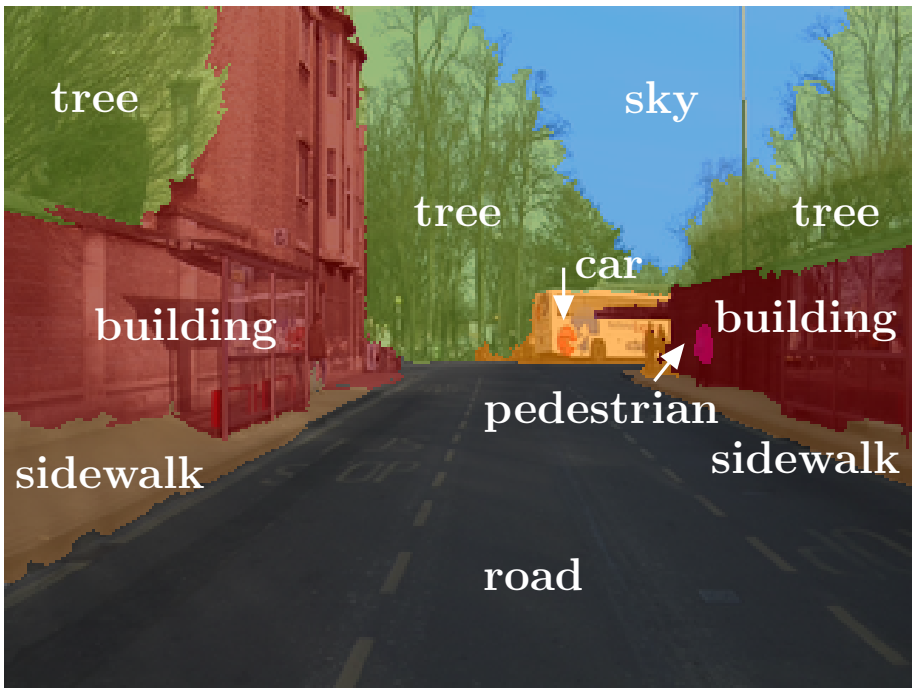



Fig. 4.1: **Total scene understanding – output of proposed method**: given an input image, the output of the system are semantic labels (sky, tree, building, road, etc.) that are consistent in both space and time (best viewed in color).

On the other hand, information about road/non-road regions are not enough for more advanced systems – let us mention self-driving cars as a great example for all of them. Moreover, all models in the self-supervised learning setup are learned online. Thus, it is difficult to validate the outputs of the system. Last, but not least, although the models are updated at each frame over the time, the output is not consistent either in space (we take all pixels that globally match to the model and are not removed by flood-fill during the post-processing), nor in time (the road/non-road regions can change their shapes in any possible manner from frame to frame).

The vision systems for more advanced applications should provide: (1) more reliable predictions (2) consistent (stable) in both, space and time and (3) information about the semantic classes present in the scene (c.f. Fig. 4.1), that can be roughly divided into two groups – objects (cars, pedestrians, etc.) and stuff (sky, grass, etc.).

The above mentioned objective of dynamic scene understanding can be formalized in the following way: given a sequence of uncalibrated monocular images $\mathcal{I} = \{\mathbf{i}^{(1)}, \mathbf{i}^{(2)}, \ldots, \mathbf{i}^{(n)}\}$ and corresponding random variables over data $\mathbf{i}^{(t)} = \{\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, \ldots, \mathbf{x}_m^{(t)}\}$, where $\mathbf{x}$ represents an image pixel, $n$ is the length of sequence and $m$ is the image size ($m = \mathit{width} \times \mathit{height}$); the labeling problem is to assign a unique label $l_i$ from the discrete set of all possible labels $\mathcal{L} = \{l_1, l_2, \ldots, l_k\}$, where $k$ is the dimension of the finite label alphabet $\mathcal{L}$. Label $l_i$ represents a sample or a "crisp output", which corresponds to the random variable over the labels for a given image $\mathbf{Y}_i^{(t)} = \{y_{i,1}^{(t)}, y_{i,2}^{(t)}, \ldots, y_{i,k}^{(t)}\}$ with the highest probability

$$l_i^{(t)} = \underset{c \in <1,k>}{\arg\max}\, \mathbf{Y}_{i,c}^{(t)}, \tag{4.1}$$

and random variables over the labels $\mathbf{Y}$ are consistent in both, space and time. It should be highlighted that pixel-wise labeling means assigning of a label to the each single pixel in the image, in contrast to object detectors, that usually provide only bounding boxes.

As we have already discussed in section 2.2, many approaches to the labeling problem exist (MRF/CRF, SHIM, ...), however most of them are consistent only in space, but not in time domain. Consequently, the labels $l_i$ are not consistent across the adjacent frames in a video stream and tend to the flickering effects. Moreover, inference does not benefit from temporal information, which might be helpful for obtaining better predictions when they are uncertain.

## 4.2 Stacked Hierarchical Inference Machines

We decided to build on an approach proposed by Munoz et al. [59] for several reasons:

1. Hierarchical inference machines have proven to be a very efficient alternative to the commonly used MRF/CRF since it mitigates both the theoretical and empirical difficulties of learning probabilistic models when exact inference is intractable.

2. Are considered to be one of the most efficient approaches among the state-of-the-art methods in both, accuracy and computational complexity.

3. Never make hard decisions and always predict a distribution of labels, so that it is possible to extract a notion of confidence in the labeling.

4. Extended version for point clouds exists [94], which promises relatively straightforward extension of our system in the future.

Stacked hierarchical inference machine proposed by Munoz et al. [59] can be described in the following way: given an input image and its hierarchical region representation a series of classifiers are trained from coarse to fine, to predict the label proportions in each region in the level. After a level has been trained, the predicted labels are passed to the child regions to be used as features that model contextual relationships. This approach is robust to the quality of the segmentation at each level as it explicitly models that regions may contain multiple labels.

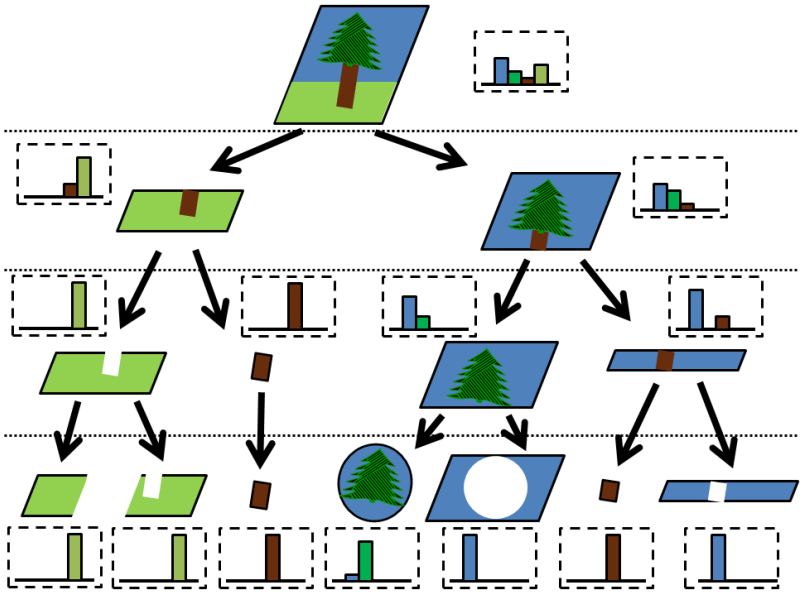
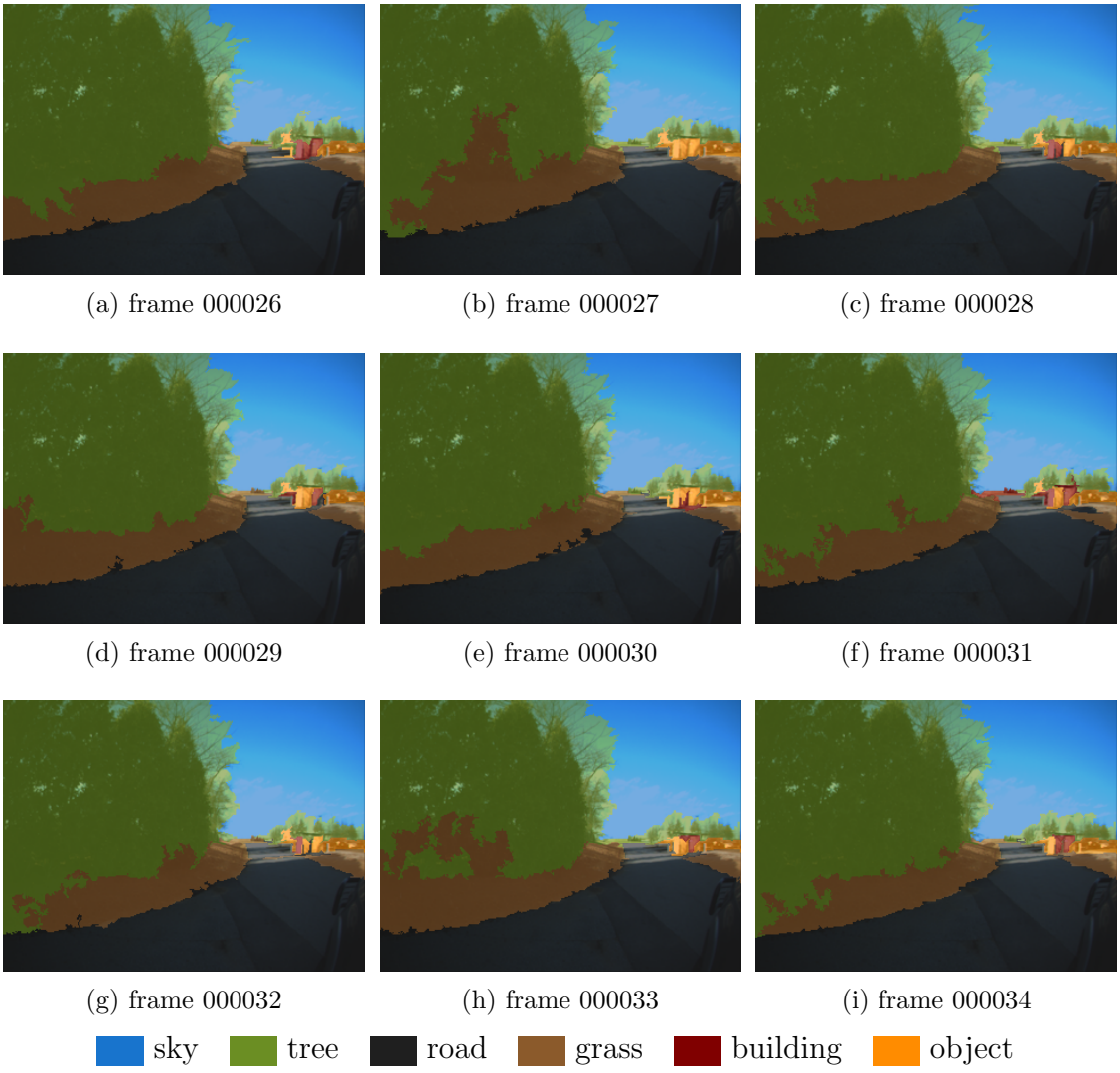


Fig. 4.2: **Stacked Hierarchical Inference Machines**: a synthetic example of a stacked hierarchical labeling process, courtesy of [59].

Although this approach is considered to be one of the best among the state-of-the-art methods, its main drawback is, that all predictions are done on single still images, i.e. it is inconvenient to use such approach for a video stream, since all the predictions in frame $t$ are completely independent on predictions from the frame $t-1$.

Hence, the output video stream tends to flicker. Although the best solution might be to completely modify the inference and consider temporal information during this stage, we decided not to change the hierarchical inference machine, because some flickering effects would be present even with the full inference. Instead, we propose a temporal smoothing filter, which reduces most flickers during the post-processing stage and can be easily combined with any other system for pixel-wise scene labeling that predicts a distribution of labels.




(a) frame 000026 (b) frame 000027 (c) frame 000028

(d) frame 000029 (e) frame 000030 (f) frame 000031

(g) frame 000032 (h) frame 000033 (i) frame 000034

Fig. 4.3: **Flickering effects** on the boundary between the trees and grass.

## 4.3 Overview

Although we build on a system proposed by Munoz et al. [59], the proposed system for spatio-temporal video labeling is a general framework that can be used with any other system which provides label distributions for all pixels in the image. Our objective can be viewed as a noise reduction in the current measurement, which is represented by an output at frame $t$.

Unfortunately, the solution is not as simple as one would expect – despite the most straightforward idea of naive averaging of predictions at corresponding pixels is close to the optimal solution, it cannot be used since we are interested in dynamic scenes understanding. The issue is, that everything in the scene is moving somehow, moreover scenes contain objects whose motion is completely independent of stuff. Thus, the pixels at the same spatial coordinates $\mathbf{x}^{(t)}[x, y]$ at frame $t$ does not correspond with the pixel at the same coordinates $\mathbf{x}^{(t-1)}[x, y]$ at frame $t - 1$.

(a) (b)

Fig. 4.4: **Scene flow**: (a) overlayed frames $t$ and $t+1$, (b) color-coded visualization of a dense optical flow estimation which demonstrates various motions in the scene.

The problem of matching corresponding pixels across the frames of a video stream is not a trivial task. We have done many experiments with naive or weighted averaging in a small neighborhood at frame $t-1$, extending of a standard FH segmentation [20] into the time dimension, or feature descriptors matching, however, none of these solutions work properly. This is illustrated in particular in the section A.

We have decided to use recently proposed Large Displacement Optical Flow (LDOF) by Brox and Malik [12] to capture the motion of image pixels. Although it was reported that LDOF provides the state-of-the-art optical flow estimation, it has errors (e.g. caused by occlusions, rounding errors, etc.) – thus, we use averaging

Fig. 4.5: **Temporal consistency**: velocity vector propagates coordinates from frame $t$ (right) to frame $t-1$ (left), where a small neighborhood is established to find the correspondences (cells are magnified for viewing purpose, they correspond to pixels in real).

in a small neighborhood at frame $t-1$ to allow some inaccuracies in optical flow estimation to overcome this problem (Fig. 4.5). More details are given in section 4.4.

The reason, why we use weighted averaging for predictions update is that there might be pixels belonging to various semantic classes in our scanning window. Hence we need to take into account only pixels belonging to the same semantic class, otherwise they would be corrupted by predictions belonging to the other semantic class. In fact, it is not important if only a single pixel or all pixels in a window at frame $t-1$ match to the reference pixel in $t$ (consider, e.g. some narrow objects like lamps, ...). We just need to ensure that the weighted average is not influenced by non-matching pixels. The main problem is how to distinguish the pixels belonging to the same class and those that do not. This issue is addressed by newly introduced learned similarity metric in section 4.5.

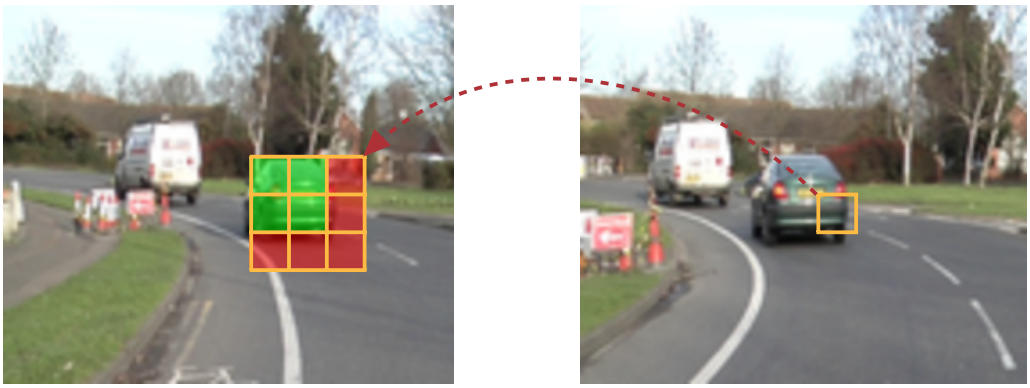


Fig. 4.6: **Matching pixels**: not all pixels (red cells) in a neighborhood are matches (cells are magnified for viewing purpose; they correspond to pixels in real).

## 4.4 Optical Flow

As we have already shown in the Fig. 4.4 and discussed in section 4.3, everything is moving somehow in dynamic scenes. We have decided not to make any assumptions or constraints about the scene. Of course, it is possible to estimate ego-motion of a camera, model the camera geometry, etc. However, we aim at designing of an easy-to-use system with no extra setup or calibration.

Although we did some experiments with varying size of the averaging window in the frame $t-1$, this approach is not correct since we assume only small motions and the difficulty of matching increases as well as computational complexity. Moreover, it is difficult to setup correctly the weights for predictions update since only a few matching pixels can be as good as all matching pixels for correct update. It is easy to show, that many situations exist when this approach fails due to the large displacement between the frames. Typical situations include camera rotation, lower frame-rate or higher speed of the vehicle. Another great example comes from camera geometry itself, objects closer to the camera makes much larger displacements than objects farther away.

(a) (b) (c)

Fig. 4.7: **Scene motion – large displacements**: (a) and (b) are input images, while (c) is the absolute value of their difference. Some parts of the images are displaced by 80 pixels.

Since we are interested in development of a general algorithm without any specific assumptions or constraints, we have decided to capture the motion by Large Displacement Optical Flow (LDOF) [12]. The most important reason why we use LDOF is its robustness to large displacements. The other commonly used optical flow estimation algorithms usually work well just for a small motion, however are not able to capture large displacements. Perhaps we should mention SIFT-flow [50] as an exception, however, it was reported that LDOF is up to 5× faster than SIFT-flow.

Moreover, GPU-based implementation [80], which is up to 80× faster than original implementation of LDOF and allows real-time processing, is available. Another important reason why we use LDOF is that the provided optical flow is dense.

### 4.4.1 Predictions Propagation

Next, we need to warp the image data $\mathbf{X}^{(t-1)}$ and predictions $\mathbf{Y}^{(t-1)}$ from the frame $t-1$ to the frame $t$. Although the LDOF is considered to be the state-of-the-art optical flow estimator (in term of accuracy), it might happen that some velocity vectors do not match exactly (e.g. rounding errors [1]) and so there could be some missing pixels, etc. As we have already mentioned in the overview, our method allows small inaccuracies in optical flow estimation. Hence, we rather warp the image and prediction from $t-1$ to $t$ by small patches instead of pixels to have as large warped area as possible without any interpolation of the data.

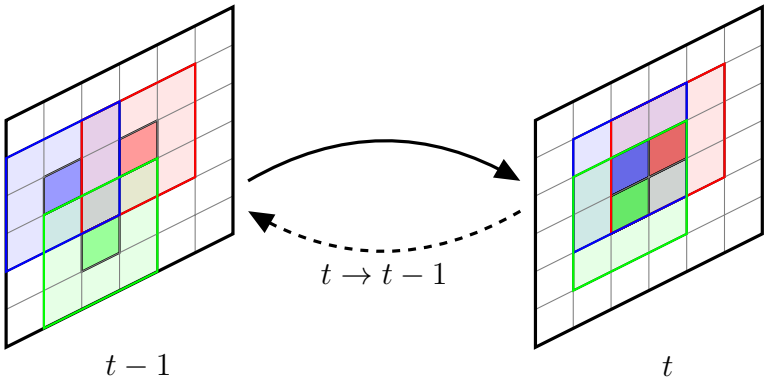



Fig. 4.8: **Predictions propagation**: warping by patches from the frame $t-1$ to frame $t$ – small filled rectangles represent pixels and flow vectors from frame $t$ to $t-1$. The larger filled areas represent the patches.

The optical flow is estimated from frame $t$ to frame $t-1$. Let $[x_t, y_t]^{\mathrm{T}}$ be a pixel at coordinates $x$, $y$ in the current frame $t$ and $\mathbf{w} = [u, v]^{\mathrm{T}}$ be an optical flow field from frame $t-1$ to $t$. The coordinates of each pixel with known velocity vector are projected from frame $t$ to frame $t-1$

$$\begin{bmatrix} x_{t-1} \\ y_{t-1} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \end{bmatrix} + \begin{bmatrix} u(x_t, y_t) \\ v(x_t, y_t) \end{bmatrix} \tag{4.2}$$

Next, warping by patches back from frame $t-1$ to $t$ can be described in the following way: for each projected pixel, establish a small region of $n \times n$ pixels

[1] LDOF provides subpixel precision, since the optical flow is estimated in a continuous domain, however both, input images and predictions are aligned in a discrete grid. It is possible to use, e.g. bilinear interpolation to deal with, however it is not necessary for our method.

around it in the frame $t-1$, project this region back to the frame $t$ and add it to the previously warped values at the same coordinates. Either image data $\mathbf{X}^{(t-1)}$ and predictions $\mathbf{Y}^{(t-1)}$ are warped in the same manner to the frame $t$, the only difference is that in the first case we project $RGB$ components of the data, while in the latter case all $k$ probabilities. Finally, warped components are independently normalized by the number of projections, so the characters of probabilities and image, respectively, are preserved.

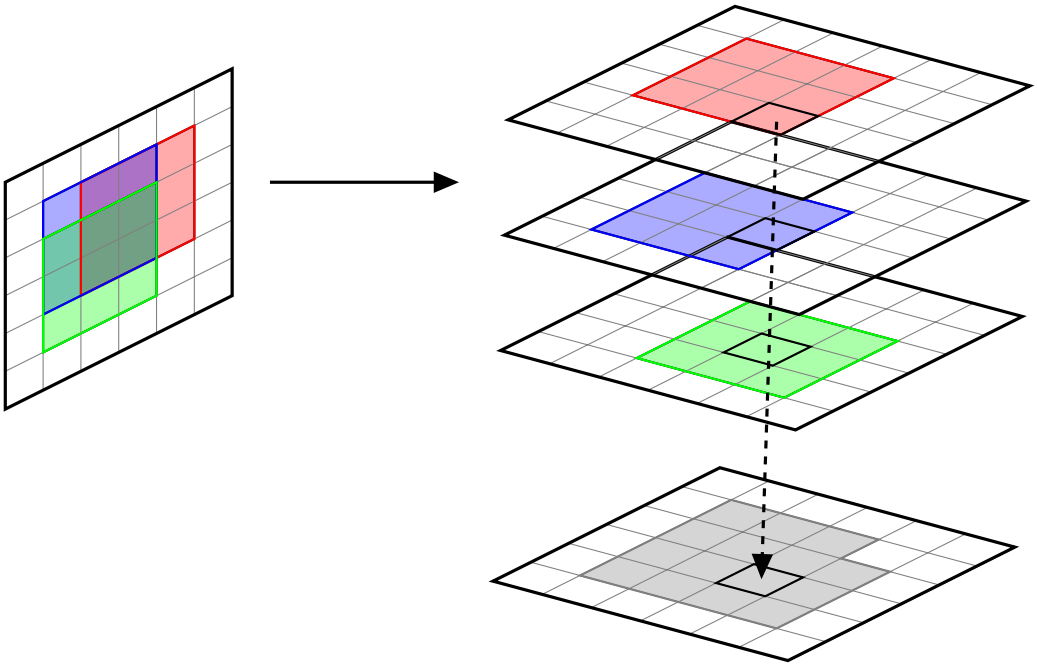


Fig. 4.9: **Normalization**: warped data (left) are independently normalized by pixels so that the results still represent image pixels or labels probabilities.

### Incremental Propagation

Let us note that in some cases it is necessary to align all images from a time window instead of just the frame $t-1$ with the reference frame $t$. Then, it makes no sense to estimate optical flow between the reference frame $t$ and all other frames $t-1$, $t-2$, $\ldots t-s$, since such pairwise flow fields need to be estimated for the each time window. A better option is to warp the information in an incremental way, so that the velocity vector for a given point from frame $t$ to $t-s$ is estimated as

$$\begin{bmatrix} u_{t\to(t-s)} \\ v_{t\to(t-s)} \end{bmatrix} = \sum_{i=0}^{i=s-1} \begin{bmatrix} u_{(t-i)\to(t-i-1)} \\ v_{(t-i)\to(t-i-1)} \end{bmatrix} \tag{4.3}$$

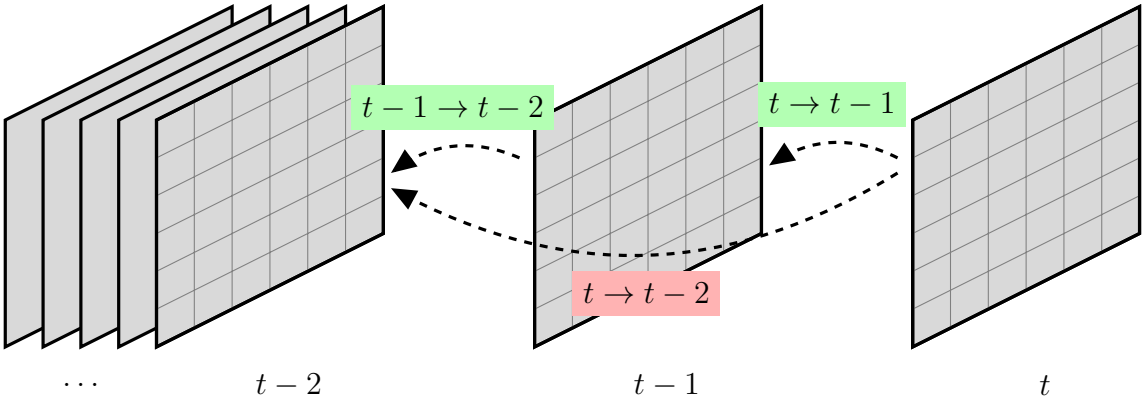



Fig. 4.10: **Incremental propagation**: it is better to use incremental propagation $t \to t-1 \to t-2$ instead of the additional estimation of an optical flow $t \to t-2$, since we have all estimates of optical flow between the adjacent frames, however all other flows need to be estimated again during the each iteration.

### 4.4.2 Forward-Backward Error

The most typical situations when the optical flow estimation fails are occlusion or incorrect estimation. If such situation occurs a reasonable demand is to stop smoothing immediately and used only a current measurement. Otherwise, we propagate pixels and predictions that might belong to the completely different parts of an image.

Commonly used approach to the estimation of optical flow failures is so-called *forward-backward error*. Such technique has proven to be efficient in tracking of objects [33] or point trajectories [80]. The idea behind this concept is simple – if we estimate optical flow from frame $t$ to $t-1$ and then the flow from this point in the frame $t-1$ back to $t$, we should get back to the original coordinates. Let forward and backward flows be

$$\mathbf{w} = \begin{bmatrix} u_{t\to(t-1)}(x_t, y_t) \\ v_{t\to(t-1)}(x_t, y_t) \end{bmatrix}, \qquad \hat{\mathbf{w}} = \begin{bmatrix} u_{(t-1)\to t}(x_{(t-1)}, y_{(t-1)}) \\ v_{(t-1)\to t}(x_{(t-1)}, y_{(t-1)}) \end{bmatrix}. \tag{4.4}$$

In the case of some inconsistency between these two velocity vectors, the estimated optical flow is not reliable for a given point. A soft criterion with tolerance interval which increases linearly with the magnitude of the motion vector is usually used ($e_1 = 0.01$, $e_2 = 0.05$)

$$|\mathbf{w} + \hat{\mathbf{w}}|^2 < e_1(|\mathbf{w}|^2 + |\hat{\mathbf{w}}|^2) + e_2, \tag{4.5}$$

however, we slightly changed this criterion

$$|\mathbf{w} + \hat{\mathbf{w}}|^2 < \kappa, \tag{4.6}$$

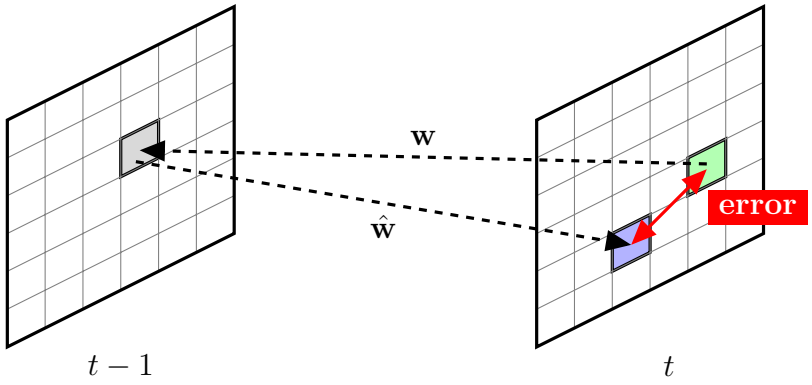
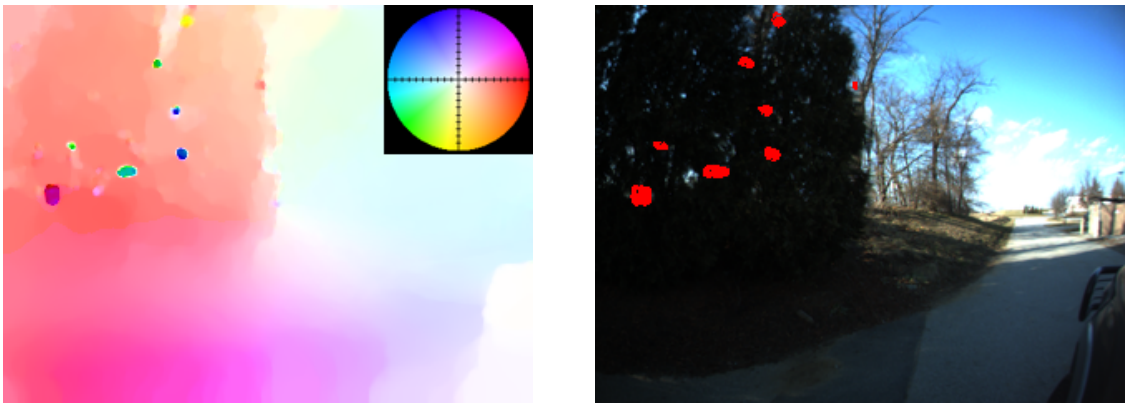



Fig. 4.11: **Forward-Backward error – illustration**: forward velocity vector projects the reference pixel (green) from frame $t$ to the frame $t-1$. However, backward optical flow projects gray pixel from $t-1$ to the frame $t$ at different coordinates. The distance between them corresponds to the Forward-Backward error.

where $\kappa$ corresponds to the size of the averaging neighborhood since this neighborhood allows inaccuracies.

Unfortunately, forward-backward error does not detect failures when both, the forward and backward flows are consistent, however e.g. the magnitude of estimated vector is smaller than the real displacement. Detection of such failures is still an open question.



(a) (b)

Fig. 4.12: **Forward-Backward error**: (a) estimated motion flow, (b) visualization of flow corruptions detected by Forward-Backward error (in red).

## 4.5 Learning Similarity Metric

Once we have warped the data from frame $t-1$ to frame $t$, most of them (excluding those with unknown or incorrect optical flow) are more or less at the corresponding spatial coordinates with data from the current measurement in the frame $t$. Now, we need to find the correspondences. Fortunately, the whole task is simplified by matching in a small, local neighborhood, however we still need to find robust descriptor and matching strategy.

### 4.5.1 Motivation

A reasonable step is to measure the similarity between the reference pixel and pixels in a small neighborhood established in the warped data, and use the measured similarities as corresponding weights for weighted averaging. We use the standard radial basis function (RBF) kernel to express the similarity between the features $\mathbf{f}_i$ and $\mathbf{f}_j$

$$w_{i,j} = e^{-\frac{d(\mathbf{f}_i,\mathbf{f}_j)^2}{\sigma^2}} \tag{4.7}$$

where $\sigma$ determines the spread of the RBF kernel; that is how quickly the function declines as the distance increased from the point.

The question is which features and which metric are suitable? Typically used color features and squared Euclidean distance $d(\mathbf{f}_i, \mathbf{f}_j) = (\mathbf{f}_i, \mathbf{f}_j)^{\mathrm{T}}(\mathbf{f}_i, \mathbf{f}_j)$ are not sufficient to distinguish small objects from the background, since small objects are usually slightly blurred (see Fig. 4.13).


(a) (b) (c)

Fig. 4.13: **Color similarity**: (a) input image, (b) zoomed in and reference pixel, (c) RBF kernel with Euclidean distance is not able to distinguish object and sky.

Hence, we need to find a better feature representation. Various features can be used, e.g. local binary patterns, gradient features, etc., however we need to find a

way to combine these features together. The problem is that each feature descriptor is sensitive to different parts of the image (illumination change, gradient, ...) and has various length (e.g. color – 3 dimensions, texture – 17, local binary patterns – 121, ...). Hence, it is very complicated to combine them together. The most straightforward way is to add more averaging terms into the update equation

$$update = \frac{1}{Z}\left[\lambda_1 T_1 + \lambda_2 T_2 + \ldots \lambda_n T_n + \lambda_c T_c\right] \tag{4.8}$$

where $\lambda$ are weights of each averaging term $T$, $T_c$ is current measurement and $Z$ is a normalizer, however it is almost impossible to find the correct weights $\lambda$. Moreover, if we would like to add another feature to our feature vector, we would need to find all the weights $\lambda$ again.

Thus, we have decided to propose a better way – instead of using so many features independently, we rather concatenate all of them together to create a new feature space and propose to use a novel similarity metric, which is based on a Mahalanobis distance parametrized by matrix $\mathbf{M}$ obtained with off-line subgradient optimization. Such approach benefits from no need to optimize many weights $\lambda$ and it is very easy to extend the feature space with another descriptors.

### 4.5.2 Training Data

We aim to obtain an $\mathbf{M}$ that results in small distances between the features that belong to the same class and a large distance between the others. Hence, we need to generate some positive and negative examples first to find the weights for the new similarity metric with subgradient optimization. Positive examples ($\mathcal{E}_p$) are correctly matching points in a given image sequence, negative examples ($\mathcal{E}_n$) are any possible distractors. We will explain this more in detail in section 4.5.4.

Of course, it is possible to take the images and start with manual clicking and pick as many matching points as we need (Fig. 4.14). A better option is to find the correct matches automatically, since we can generate as many samples as we need and easily change sample images without any time-consuming manual work.

Usually, the work-flow can be roughly described as detection of local invariant keypoints in images, extraction of descriptors and matching. Matching usually consists of two steps: searching for tentative correspondences (exhaustive search, efficient data structures, ...) and refinement (robust estimation of correct matches with RANSAC, ...).

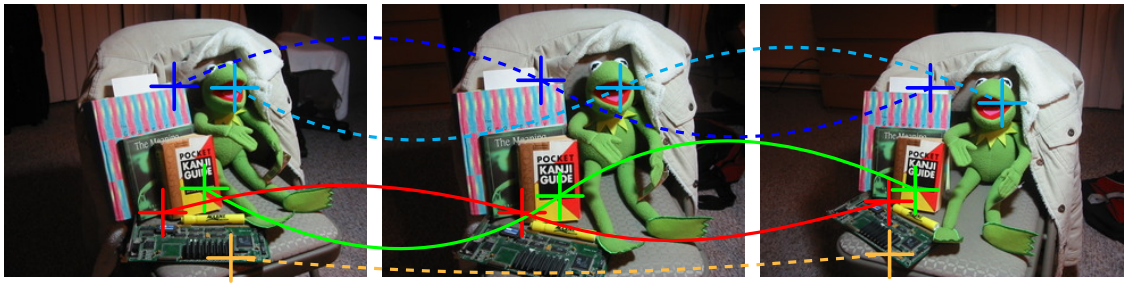


Fig. 4.14: **Correspondences**: first 5 matches in the first 3 images of Kermit sequence. The orange keypoints were found in images 1 and 3 (e.g. first row of Tab. 4.13). Combination of single-colored crosses generates positive example and mixed colors are negatives (see sec. 4.5.4).

We use publicly available package Bundler [74] – a structure-from-motion (SfM) system for unordered image collections, which is the problem of using 2D measurements arising from a set of images of the same scene in order to recover information related to the 3D geometry of the imaged scene as well as the locations and optical characteristics of the employed camera(s).

Bundler takes a set of images, image features and tentative matches as input, and produces a 3D reconstruction of camera and (sparse) scene geometry as output. The system reconstructs the scene incrementally – a few images at a time, using the Sparse Bundle Adjustment library of Lourakis and Argyros [51] which is considered to be the standard bundle adjustment implementation. Bundle adjustment is usually the last step of structure-from-motion algorithms – it is an optimization problem over the 3D structure and viewing parameters (camera pose, intrinsic calibration, & radial distortion parameters), which are simultaneously refined for minimizing reprojection error.

One can argue that using Bundler is an overshoot, since we need only correspondences, while Bundler computes even a 3D reconstruction of camera and scene geometry. This objection is true, however Bundler is easy-to-use since we need to provide only input images, features with tentative matches and parse the output.

We use Kermit sequence (Fig. 4.15), which is provided with Bundler package as an example. This sequence consists of 11 images of an indoor scene. Although the input images are completely different from our outdoor sequence, the learned similarity distance works well. More details about positive and negative examples are explained in section 4.5.4.

Fig. 4.15: **Kermit sequence**: training data are completely different from our test sequences.

### 4.5.3 Features & Preprocessing

Once we have all the correspondences, it is necessary to extract the features. We have observed, that the best trade-off between the computational complexity and and feature descriptor's richness is obtained by a combination of Local Binary Patterns (LBP)s, texture and color (RGB) features. As we have already mentioned in the state-of-the-art (sec. 2.4.3), textons and color are real-valued while LBPs are bit strings. All the descriptors are concatenated to a single column vector for each keypoint

$$\mathbf{f} = \left[ \begin{array}{cccccccccccc} LBP_1 & LBP_2 & \dots & LBP_n & txt_1 & txt_2 & \dots & txt_m & r & g & b \end{array} \right]^{\mathrm{T}} \tag{4.9}$$

where indices $n = 121$ and $m = 17$ are lengths of descriptors, so the concatenated feature vector has 141 dimensions ($D$).

Since we combine various descriptors into a single feature vector, the range of values of raw data varies widely. Hence, the next step is normalization of all features. This has two practical reasons: (1) if one of the features has a broad range of values, the distance will be governed by this particular feature and (2) it helps to a faster convergence of subgradient optimization.

Various scaling techniques can be employed, we use one of the most widely suggested – z-scores. First, we take all the feature vectors for one image, form a

$N \times D$ matrix $\mathbf{F}$, where $N$ is the number of features and $D$ is the feature dimension

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}_1^{\mathrm{T}} \\ \mathbf{f}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{f}_N^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} LBP_{11} & LBP_{12} & \dots & LBP_{1n} & txt_{11} & txt_{12} & \dots & txt_{1m} & r_1 & g_1 & b_1 \\ LBP_{21} & LBP_{22} & \dots & LBP_{2n} & txt_{21} & txt_{22} & \dots & txt_{2m} & r_2 & g_2 & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ LBP_{31} & LBP_{32} & \dots & LBP_{3n} & txt_{31} & txt_{32} & \dots & txt_{3m} & r_3 & g_3 & b_3 \end{bmatrix} \tag{4.10}$$

Then we compute the deviation scores separately within dimensions (the values in each dimension have zero mean), which can be written in a matrix form [2] as

$$\mathbf{F}_{DC} = \mathbf{F} - \mathbf{1}\mathbf{1}^{\mathrm{T}}\mathbf{F}(\mathbf{1}/N) \tag{4.11}$$

where 1 is a $N \times 1$ column vector of ones ($\mathbf{1} = [\ 1_1\ \ 1_2\ \ \dots\ \ 1_N\ ]^{\mathrm{T}}$) Next, we compute the covariance matrix

$$\mathbf{F_{cov}} = \mathbf{F}_{DC}^{\mathrm{T}}\mathbf{F}_{DC}(\mathbf{1}/N) \tag{4.12}$$

and normalize the features in $\mathbf{F}$ by the standard deviations that are the square roots of elements on the main diagonal of the covariance matrix $\mathbf{F_{cov}}$
(i.e. $[\ \sqrt{f_{cov11}}\ \ \sqrt{f_{cov22}}\ \ \dots\ \ \sqrt{f_{covdd}}\ ]$), so that the features in $\mathbf{F}$ have zero mean and unit variance. Finally, all values are truncated so that they are in the range $[-\eta, \eta]$ (we use $\eta = 5$). Standardized feature matrix is referred as $\mathbf{F}_z$ in the next parts for clarity.

### 4.5.4 Learning & Measurement

Now we can generate positive ($\mathcal{E}_p$) and negative ($\mathcal{E}_n$) examples. An example is a row feature vector $\mathbf{f}_{z,ij} = |\mathbf{f}_{z,i} - \mathbf{f}_{z,j}|$, where $|\cdot|$ is an absolute value of differences between feature dimensions $\mathbf{f}_{z,i}(d)$ and $\mathbf{f}_{z,j}(d)$. This holds for both, positive and negative examples. The only difference is in indices $i$ and $j$ for positive examples ($\mathcal{E}_p$) and $i$ and $k$ for negatives ($\mathcal{E}_n$).

The output of Bundler can be represented as a list of correspondences consisting of $\vartheta$ rows ($\vartheta$ is the number of all correspondences in a given sequence). Each row of the list consists of a number of images the point is visible in, and corresponding number of triplets consisting of an image number, x and y coordinates of that

[2] Similar equations in a form with sums instead of matrices are given in section 3.3.2

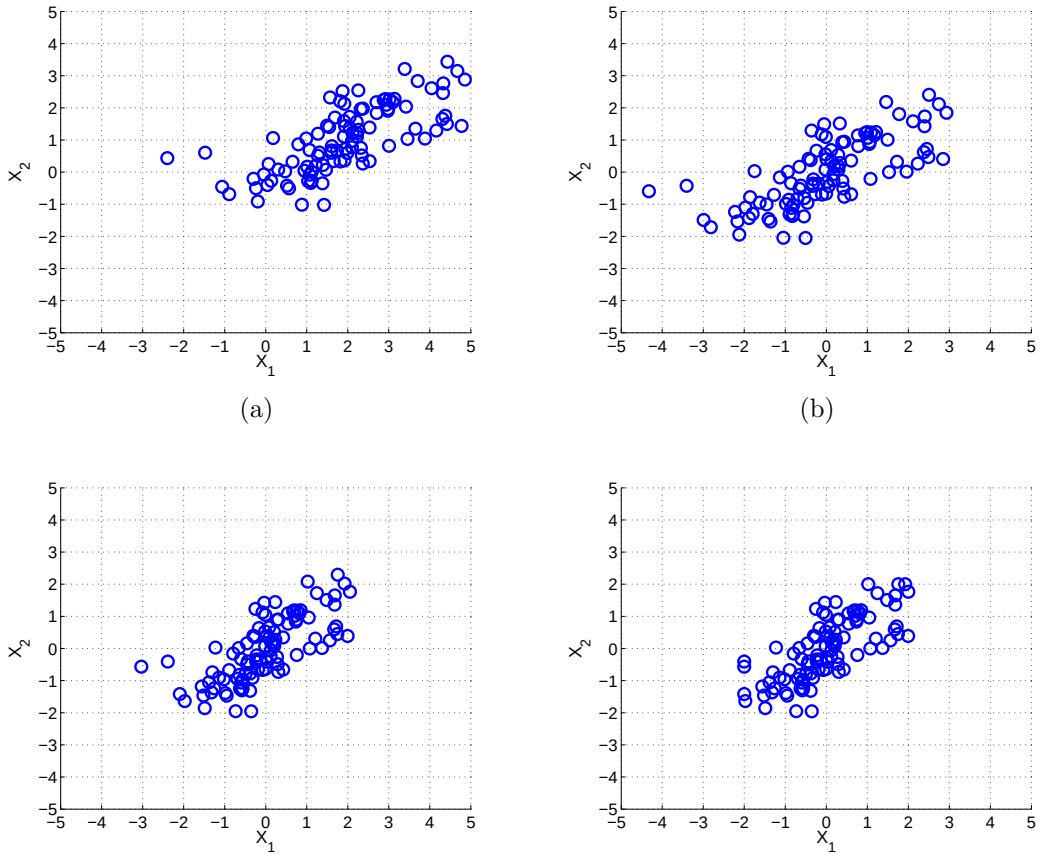



Fig. 4.16: **Feature scaling – synthetic example**: (a) original features, (b) subtracted means, (c) normalized variances, (d) truncated to [-2 2].

keypoint.

$$list = \left\{ \begin{array}{cccccccccccc} \texttt{<views>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & & & & & \\ \texttt{<views>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & \ldots & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} \\ \texttt{<views>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & & \\ \vdots & & & & & & & & & & & \\ \texttt{<views>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & \texttt{<img>} & \texttt{<x>} & \texttt{<y>} & & & & & \end{array} \right\} \tag{4.13}$$

To limit the combinatorial explosion, we consider as a positive example a set $\mathcal{E}_p$ of all possible pairwise combinations of the first and the rest of corresponding keypoints at each row. To have a balanced training data, a single negative example is generated for each positive example. Negative example is a pairwise of the first keypoint at each row and any other random keypoint from different row (non-corresponding keypoint). A set of all negative examples is $\mathcal{E}_n$.

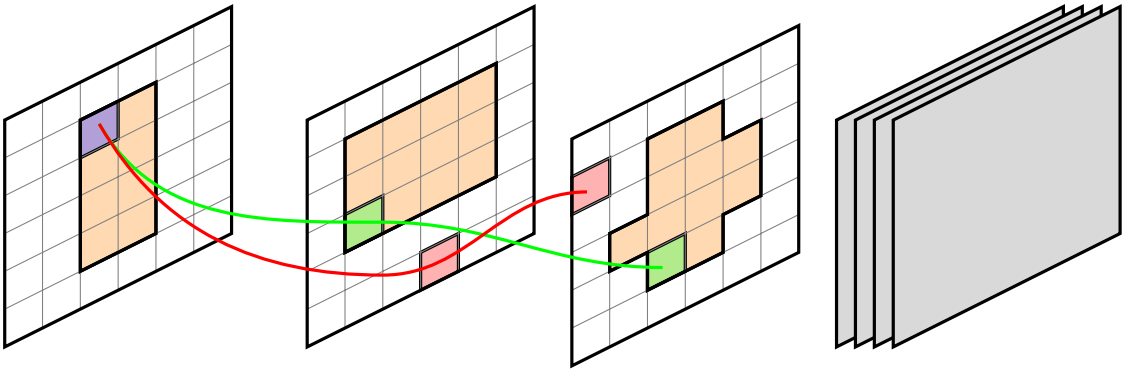


Fig. 4.17: **Training data – positive & negative examples**: illustration for the 3rd row of Eq. 4.13. The first pixel on the row is blue. The green pixels are correspondences. The positive examples are pairs of blue and green pixels from the 2nd and 3rd row, respectively. Negative examples are pairs between the blue and red pixels from the 2nd and 3rd frame. The orange areas represent the same object under multiple viewpoints.

We aim to obtain an $\mathbf{M}$ that results in a small distance ($d_M(\mathbf{f}_{z,i}, \mathbf{f}_{z,j}) \leq 1$) between the features in $(i,j) \in \mathcal{E}_p$ and a large distance ($dM(\mathbf{f}_{z,i}, \mathbf{f}_{z,k}) \geq 2$) between regions in $(i,k) \in \mathcal{E}_n$ . This optimization problem can be written as the convex problem

$$\begin{aligned} \min_{\mathbf{M},e,\epsilon} \quad & \|\mathbf{M}\|_{L2}^2 + \alpha \sum_{i,j} e_{i,j} + \beta \sum_{i,k} \epsilon_{i,k} \\ s.t. \quad & |\mathbf{f}_{z,i} - \mathbf{f}_{z,j}|^{\mathrm{T}} \, \mathbf{M} \, |\mathbf{f}_{z,i} - \mathbf{f}_{z,j}| \leq 1 + e_{ij}, \qquad \forall (ij) \in \mathcal{E}_p \\ & |\mathbf{f}_{z,i} - \mathbf{f}_{z,k}|^{\mathrm{T}} \, \mathbf{M} \, |\mathbf{f}_{z,i} - \mathbf{f}_{z,k}| \geq 2 + e_{ik}, \qquad \forall (ik) \in \mathcal{E}_n \\ & M \in \mathcal{M}, \end{aligned} \tag{4.14}$$

where $\mathcal{M} = \mathbf{M}| \preceq 0, \mathbf{M} = \mathbf{M}^{\mathrm{T}}\}$ is the convex cone of symmetric positive semidefinite matrices, and $\alpha$ and $\beta$ are terms that penalize when two pairs cannot be sufficiently close or far away, respectively. Letting $\triangle_{ij} = |\mathbf{f}_{z,i} - \mathbf{f}_{z,j}|^{\mathrm{T}}$ , then the program can be rewritten as

$$\min_{\mathbf{M} \in \mathcal{M}} \mathrm{tr}(\mathbf{M}^{\mathrm{T}}\mathbf{M}) + \alpha \sum_{ij} \max(0, \mathrm{tr}(\mathbf{M}^{\mathrm{T}} \triangle_{ij}) - 1) + \beta \sum_{ij} \max(0, 2 - \mathrm{tr}(\mathbf{M}^{\mathrm{T}} \triangle_{ik})) \tag{4.15}$$

Optimizing Eq. 4.15 with the subgradient method can be used to obtain the global solution. We define$\mu_{ij}$ and $\nu_{ij}$ to be subgradients of the terms in the first and second summation, respectively:

$$\mu_{ij} = \begin{cases} \triangle_{ij}, & \mathrm{tr}(\mathbf{M}^{\mathrm{T}} \triangle_{ij}) - 1) > 0 \\ 0 & \text{otherwise} \end{cases}, \quad \nu_{ij} = \begin{cases} -\triangle_{ik}, & 2 - \mathrm{tr}(\mathbf{M}^{\mathrm{T}} \triangle_{ij})) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.16}$$

The update rule with some step-size $\eta_t$ is then

$$\mathbf{M}_{t+1} \leftarrow \mathcal{P}_{\mathcal{M}} \left( \mathbf{M}_t - \eta_t \left( \mathbf{M}_t + \alpha \sum_{ij} \mu_{ij} + \beta \sum_{ij} \nu_{ik} \right) \right), \tag{4.17}$$

where $\mathcal{P}_{\mathcal{M}}$ is an operator that projects the matrix to the closest symmetric posi-semidefinite matrix. In practice this is done by examining the eigen-decomposition of the matrix and then setting the negative eigenvalues to 0.

Once we have learned the similarity distance (off-line), on-line measurements between the reference $\mathbf{f}_{z,r}$ and query features $\mathbf{f}_{z,q}$ are standard general inner products

$$d_{sym}(\mathbf{f}_{z,r}, \mathbf{f}_{z,q}) = |\mathbf{f}_{z,r} - \mathbf{f}_{z,q}|^{\mathrm{T}} \, \mathbf{M} \, |\mathbf{f}_{z,r} - \mathbf{f}_{z,q}|, \tag{4.18}$$

where $\mathbf{M}$ are weights of the learned similarity metric.

## 4.6 Temporal Smoothing

Now we have all the necessary prerequisites for temporal smoothing. The temporal smoothing starts with optical flow estimation discussed in section 4.4 and continues with update introduced in this section. Although we had to make some effort to train the similarity distance, now we will benefit from it since we got rid of all the difficulties with combination of various averaging terms and the whole update is reduced into a single and simple equation. Moreover, the distance is learned just once during the training stage and can be used on various testing sequences.

The proposed recursive temporal smoothing can be described in a following way: given a reference frame at time $t$, frame warped from time $t-1$, normalized features $\mathbf{F}_z$ computed on image data $\mathbf{i}$ and predictions $\mathbf{Y}$, update all the pixels from $t$ with known optical flow according to the following equation

$$\hat{\mathbf{Y}}_i^{(t)} = \frac{1}{Z} \left[ \sum_{j \in N_i} w_{ij} \hat{\mathbf{Y}}_j^{(t-1)} + c \mathbf{Y}_i^{(t)} \right], \tag{4.19}$$

where $\hat{\mathbf{Y}}_i^{(t)}$ are smooth label probabilities for a pixel $i$, $N_i$ is a local neighborhood (usually $5 \times 5$ pixels) in warped random variables over the labels, $\mathbf{Y}_i$ are raw probabilities for a given pixel $i$, $c$ is the learning rate ($c = 0.25$) and $Z$ normalizes the smoothed predictions so that they still represent probabilities. Similarity weights $w_{ij}$ correspond to the radial basis function (RBF) kernel with learned similarity metric

$$w_{ij} = e^{-\frac{d_{sym}\left(\mathbf{f}_{z,i}^{(t)}, \mathbf{f}_{z,j}^{(t-1)}\right)^2}{\sigma^2}}, \tag{4.20}$$

where $\mathbf{f}_{z,i}^{(t)}$ is a normalized feature vector for reference pixel $i$ in frame $t$, $\mathbf{f}_{z,j}^{(t-1)}$ is a normalized feature vector for warped matching pixel $j$ and $\sigma$ determines the spread of the RBF ($\sigma = 0.3D$).

If the environment is not too much challenging for reliable optical flow estimation, it is possible to use an alternative update equation

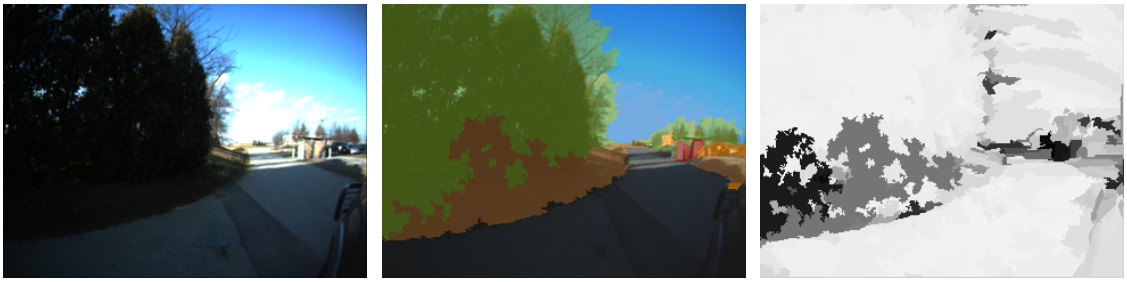
$$\hat{\mathbf{Y}}_i^{(t)} = (1-\lambda)\frac{1}{Z_1}\left[\sum_{j\in N_i} w_{ij}\hat{\mathbf{Y}}_j^{(t-1)} + c\mathbf{Y}_i^{(t)}\right] + \lambda\frac{1}{Z_2}\left[\sum_{j\in<t-s_1,t+s_2>} w_{ij}\mathbf{Y}_i^{(j)} + c\mathbf{Y}_i^{(t)}\right], \tag{4.21}$$

where $\lambda = e^{-\frac{\sum w_i j}{|j|\cdot\sigma_t}}$, $\sigma_t$ determines spread of the RBF kernel, $s_1$ and $s_2$ define the length of a time window and $|j|$ is the number of frames in a time window. This equation produces smoother results since $\lambda$ measures the consistency of the corresponding predictions (raw) in a time window. If the raw predictions are consistent within the time window (i.e. same class label), we do not need any recursive smoothing and the second term is used. If the predictions are inconsistent, we use the first term exactly as in Eq. 4.19. The modified equation allows to use much lower weights $c$ for current measurement and allows an easy way how to control the number of frames if the predictions are considered as flickering effect or a new correct class label. The lower weights $c$ can be used since if the predictions are consistent, they remain unchanged without any recursive smoothing. Another benefit is, that such predictions are not influenced by predictions from the frame e.g. $t - 100$.

The reason why such a simple method works is that we propagate the recursive averaging through the time. However this averaging is weighted by the similarity between the reference and corresponding pixels. Thus, only the correct labels are used for update and we are able to preserve the shape of small objects.

## 4.7 Further Analysis & Confidence Score

Another benefit which comes from probabilistic processing are possibilities of further analyses. Since we propose a temporal smoothing filter, it can reduce some flickers, however it cannot improve the predictions if they are constantly wrong – typical example are pedestrians. The portion of pedestrians is lower than 1% of examples in the training database. One draw back of SHIM is, that it is learned by images, thus is is not possible to balance the data and predictions for small classes might fail. This is a typical situation which can not be solved by temporal filter.



(a) (b) (c)

Fig. 4.18: **Confidence score**: a) input image, b) output of the SHIM, c) confidence score – the brighter the area is, the higher confidence. The most inference failures are in the areas with low confidence.

The other situation is if the predictions are uncertain, e.g. two classes have probabilities of approximately 40%. The smoothing filter is usually not able to improve the predictions since these two classes have almost the same probabilities. The good news is that we are able to detect such behaviour with confidence score. We can define the confidence score as

$$conf_i = y_{i,m1} - y_{i,m2} \tag{4.22}$$

where $y_{i,mx}$ are class labels with the highest and second highest probabilities for a given pixel $i$.

Confidence score is useful since we can visualize which area can not be smoothed and thus it helps to avoid overfiting.

## 4.8 Towards Full Inference

Everything what we have mentioned so far is just a first step on a long road towards full inference in space and time. The reasons why we have started with smoothing filter are: (1) even with full inference, flickering effects would be present in an output and (2) we have been able to test various approaches to the spatio-temporal consistency and their behaviour (especially extended FH segmentation) so we have sufficient knowledge about their usability for full inference.

# 5 EXPERIMENTAL RESULTS & LESSONS LEARNED

*This chapter gives a performance evaluation of proposed methods. First, we evaluate methods from chapter 3 – a vanishing point estimation and the road extraction algorithm. The second part of this chapter discusses results for spatio-temporal consistency and smoothing filter proposed in chapter 4.*

## 5.1 Self-supervised Learning

The vanishing point estimation is tested separately from the road extraction part, on the same data set used by Kong et al. [39, 38] for two main reasons: (1) this dataset is very challenging for vanishing point estimation since it contain many sequences with roads barely visible even for humans, and (2) enables a fair comparison with preceding approaches.

Road extraction algorithm is tested on a number of different sequences, which consist from more than 10 000 images captured by Orpheus-AC because, in contrast to e.g. domain of local invariant feature detectors and descriptors, to our best knowledge no standardized dataset and performance evaluation framework exists[1], like for features does [55]. Moreover, the test sequences of previously published papers [17, 18] are not freely available.

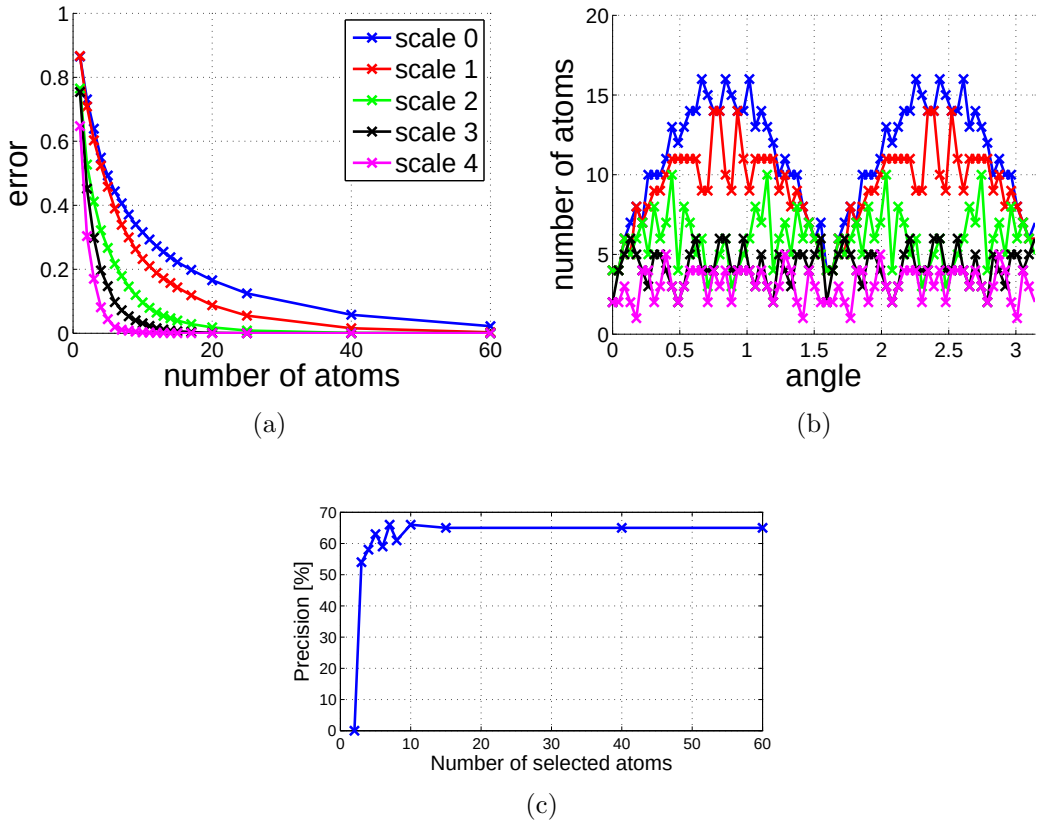
### 5.1.1 Vanishing Point Estimation

The data set consists of 1003 images. Among them, about 430 images are from photographs taken on a scouting trip along a possible Grand Challenge route in the Southern California desert and the other part is downloaded from internet by Google Image. All images are normalized to the same height and width of 128, as we have suggested in [58]. The ground truth data were obtained by manual labeling: 5 persons marked the vanishing point location, a median of these results is used as the initial ground-truth position. The two farthest manually marked locations to the initial ground-truth position are removed as outliers. Finally, the ground-truth location is computed as the mean of the other three locations. It should be noted that we compare our results against Kong et al. [39, 38] method without road segmentation, since ground-truth data for road extraction part are not freely available.

[1] The datasets such as CamVid, ... are captured in a city environment, which is not the primary environment of our algorithm since it contain many stop-and-go situations, that are not suitable for self-supervised learning (the training area is occluded by cars, ...).

### Approximation of Gabor wavelets

First of all, we evaluate how many atoms are needed to sufficiently approximate Gabor wavelets. The lower the number of approximating atoms is, the higher the speed of filtering, however, the worse approximation. We run the OOMP and set desired number of basis (varied from 1 to 60) and measure the sum of absolute differences between Gabor wavelet and the reconstructed approximation. Fig 5.1 shows this error in each scale separately. It is obvious that the precision varies over the scales. Hence, we do not choose the number of atoms, however set the precision and OOMP simply stops if the precision of approximation is below this threshold ($t = 0.35^{1+0.1s}$). This is useful since the number of selected atoms is not varied just over the scales, however even over the angles – it is clear that filters with some orientations need more basis than others.




(a) (b)

(c)

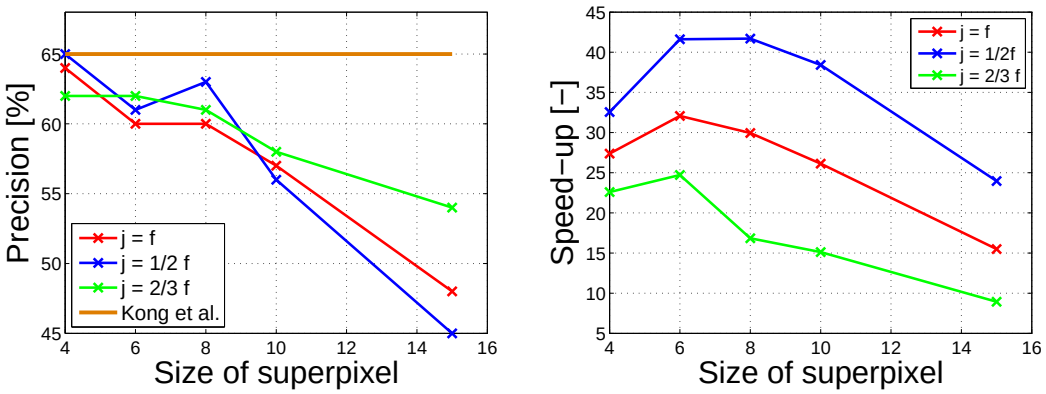
Fig. 5.1: **Evaluation of Gabor wavelets approximations**: the dependency of a number of selected basis and precision.

Figure 5.1 shows how the approximation error $\frac{\|\psi-\widehat{\psi}\|}{\|\psi\|}$ is dependent on a number of selected atoms. Each scale is approximated with different error (c.f. Fig. 5.1 a)

and each orientation of a filter require a different number of selected basis to ensure the required precision (c.f. Fig. 5.1 b). Fig. 5.1 c) compares precision (percentage of images with an estimated vanishing point within a region of 15 pixels around the ground-truth) of a various number of selected atoms. We use Kong et al. [39, 38] voting scheme for this evaluation to be independent on the quality of our voting method. Only 8 atoms are enough for reliable estimation of a vanishing point.

### Efficient voting

Next, we evaluate the precision and a speed of our voting scheme against Kong et al. [39, 38]. The speed of our voting scheme is dependent on the size of superpixels and the subregions used for refinement. Figure 5.2 shows that the best trade-off between precision and speed-up is obtained for $f = 8$ and the size of subregions $j = 1/2f$. It is important, that our voting scheme is 41.7 times faster than Kong et al. [39, 38] while we loose only 3% in precision.




(a) (b)

Fig. 5.2: **Evaluation of the vanishing point estimation**: The dependency of a precision, size of superpixels and refinement regions (a), and a speed-up of a voting scheme against Kong et al. [39, 38] (b).

### Examples

Figures 5.3 – 5.7 show typical results for desert (Fig. 5.3, 5.4), snow (Fig. 5.5, 5.6) and suburban environments (Fig. 5.7): input images (a), overlayed dominant orientations (b) and jet colormap (c), 4 levels of superpixels (d), coarse-to-fine voting (e),(f) and output images (g).

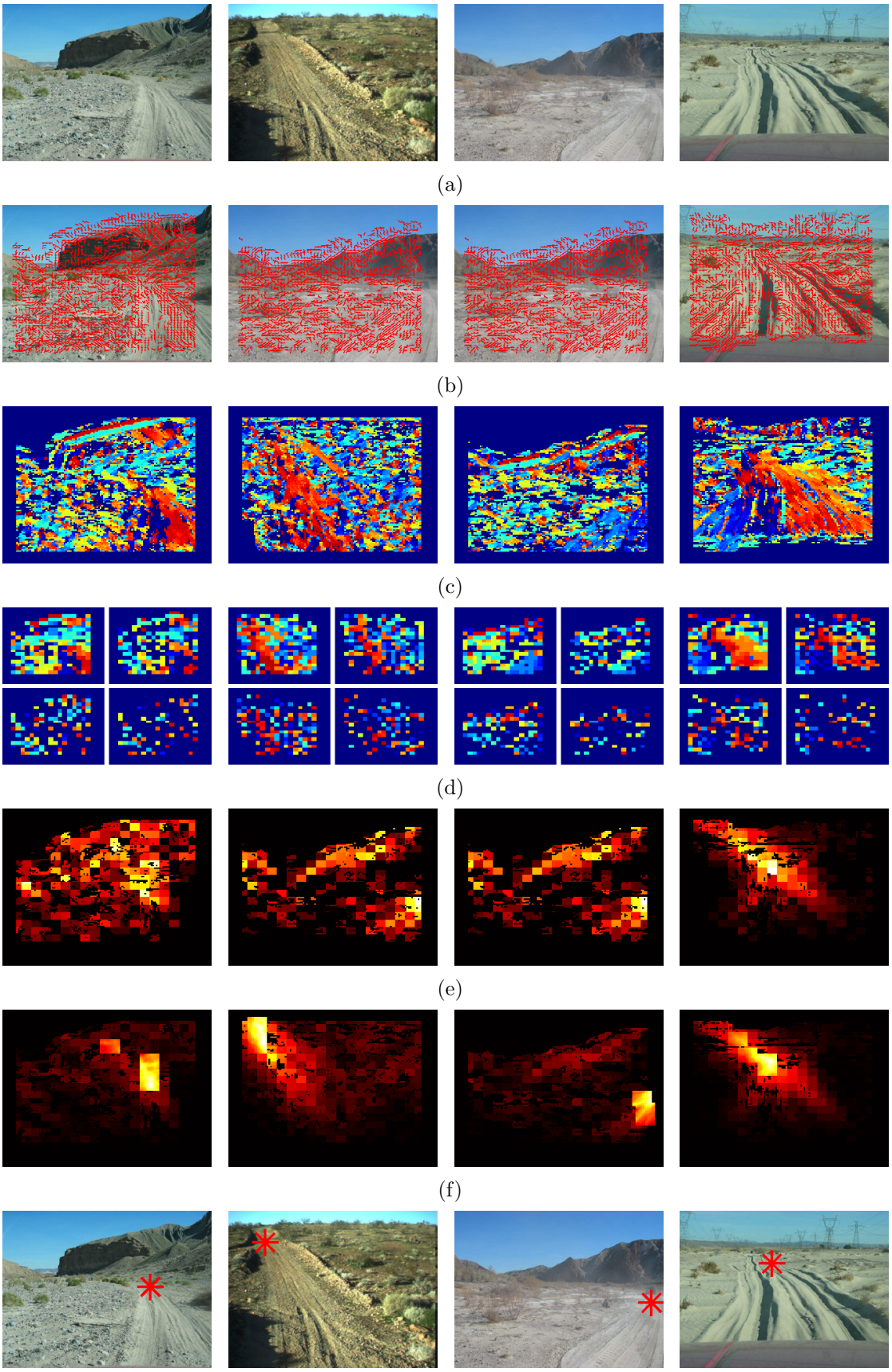


Fig. 5.3: **Vanishing point estimation**: examples - Mohave desert I. – inputs (a), dominant orientations (b,c), superpixels (d), coarse-to-fine voting (e,f), outputs (g).

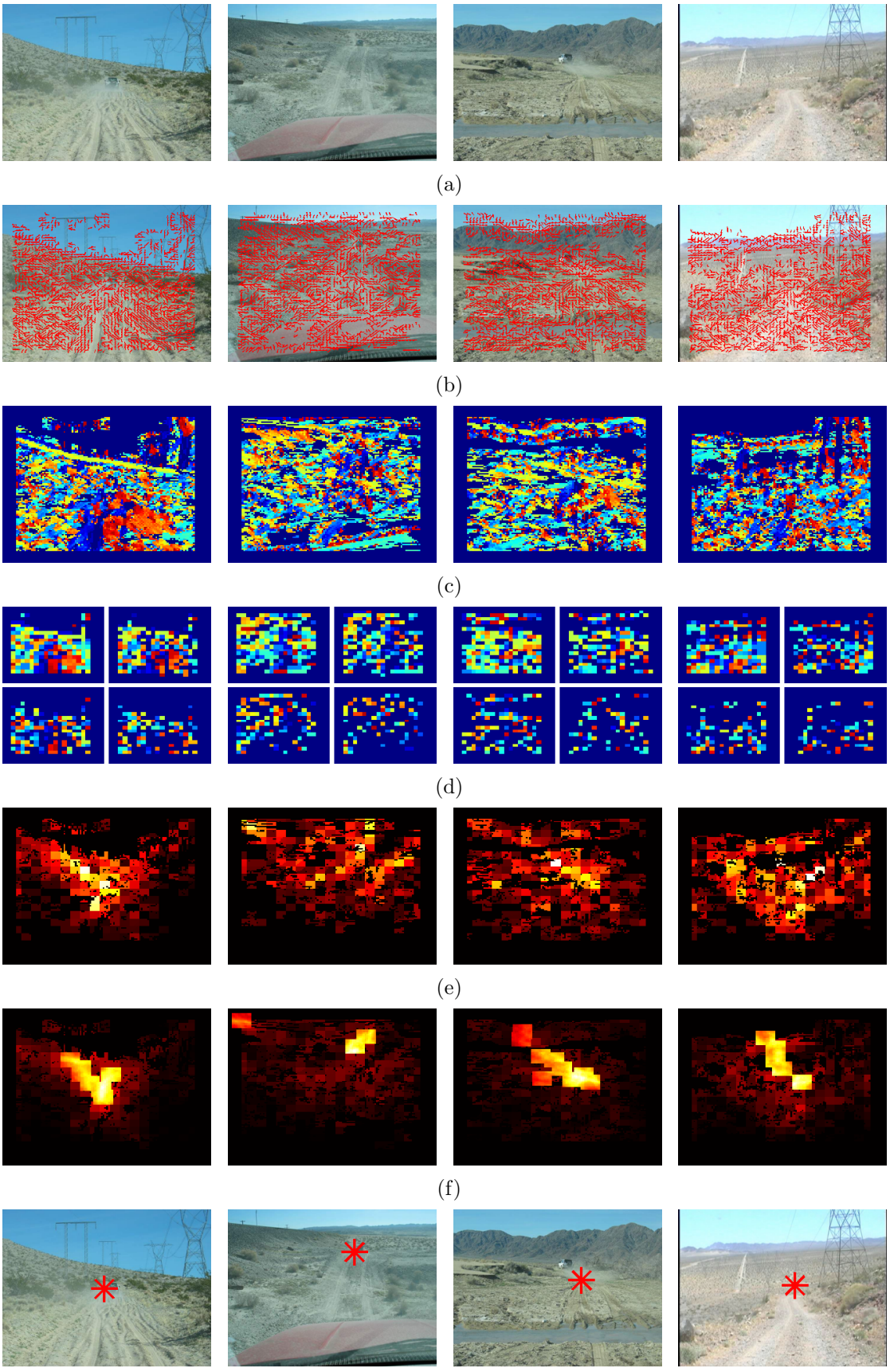


Fig. 5.4: **Vanishing point estimation**: examples – Mohave desert II. – inputs (a), dominant orientations (b,c), superpixels (d), coarse-to-fine voting (e,f), outputs (g).

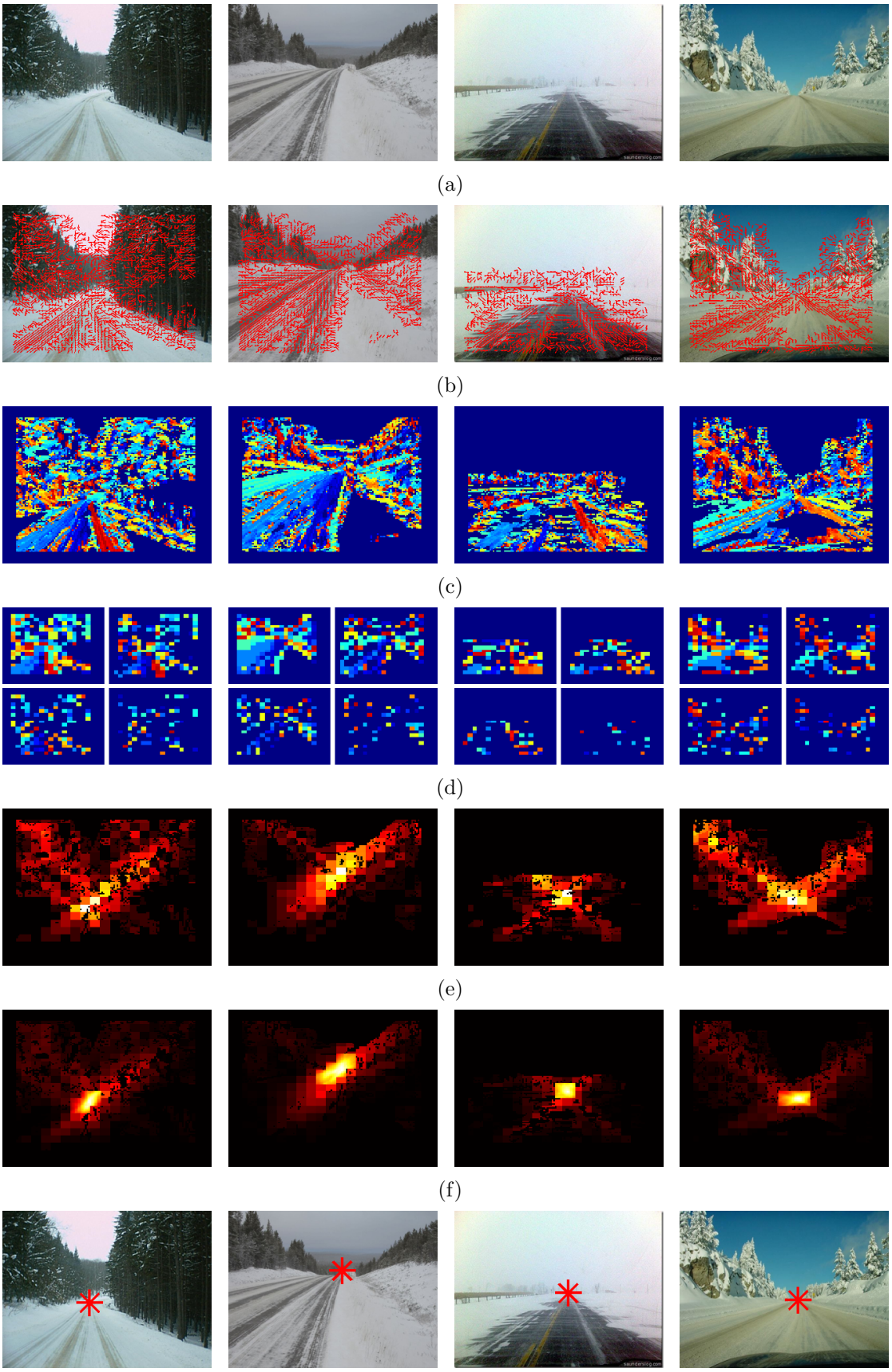


Fig. 5.5: **Vanishing point estimation**: examples – snow I. – inputs (a), dominant orientations (b,c), superpixels (d), coarse-to-fine voting (e,f), outputs (g).

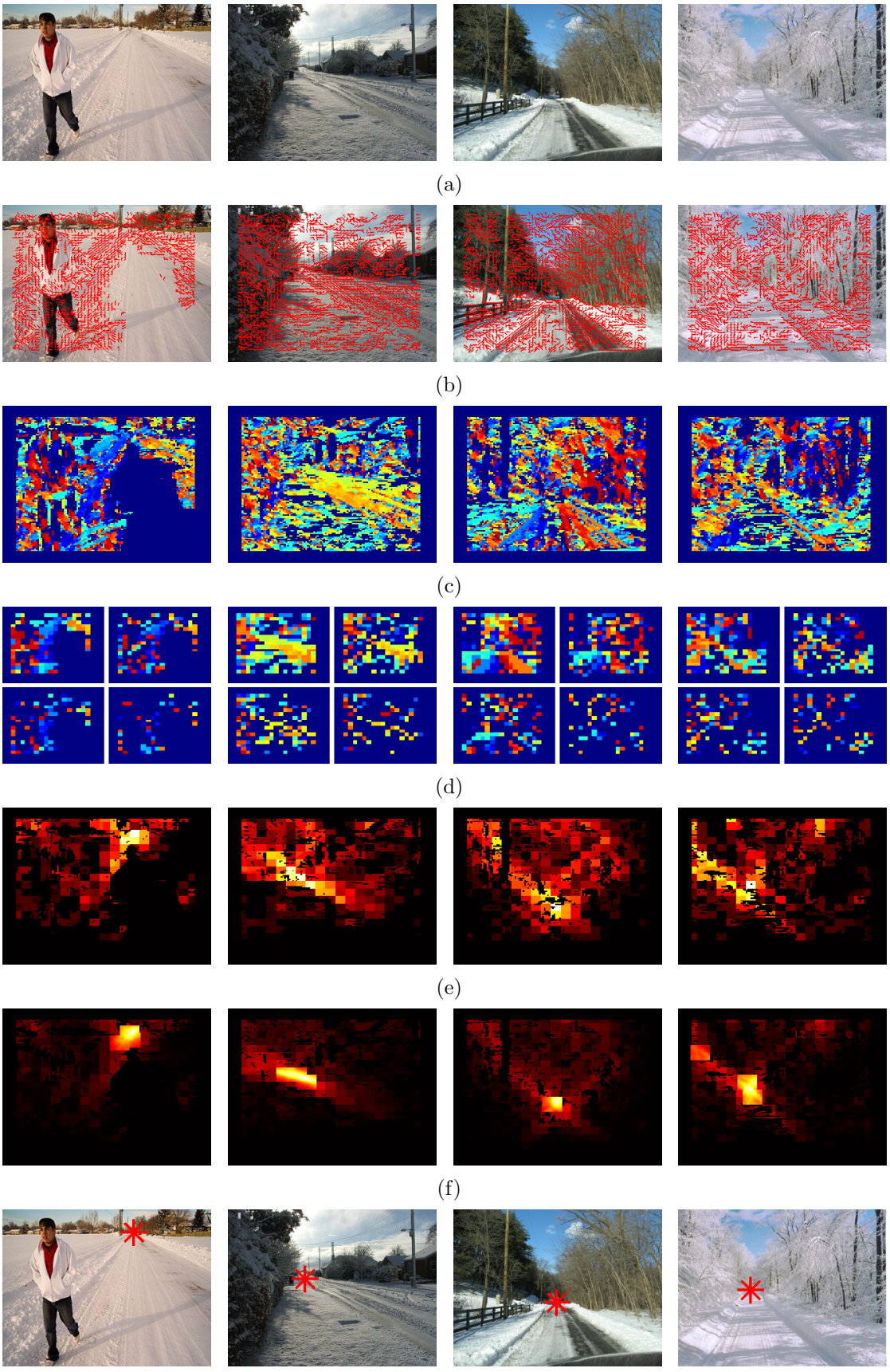


Fig. 5.6: **Vanishing point estimation**: examples – snow II. – inputs (a), dominant orientations (b,c), superpixels (d), coarse-to-fine voting (e,f), outputs (g).

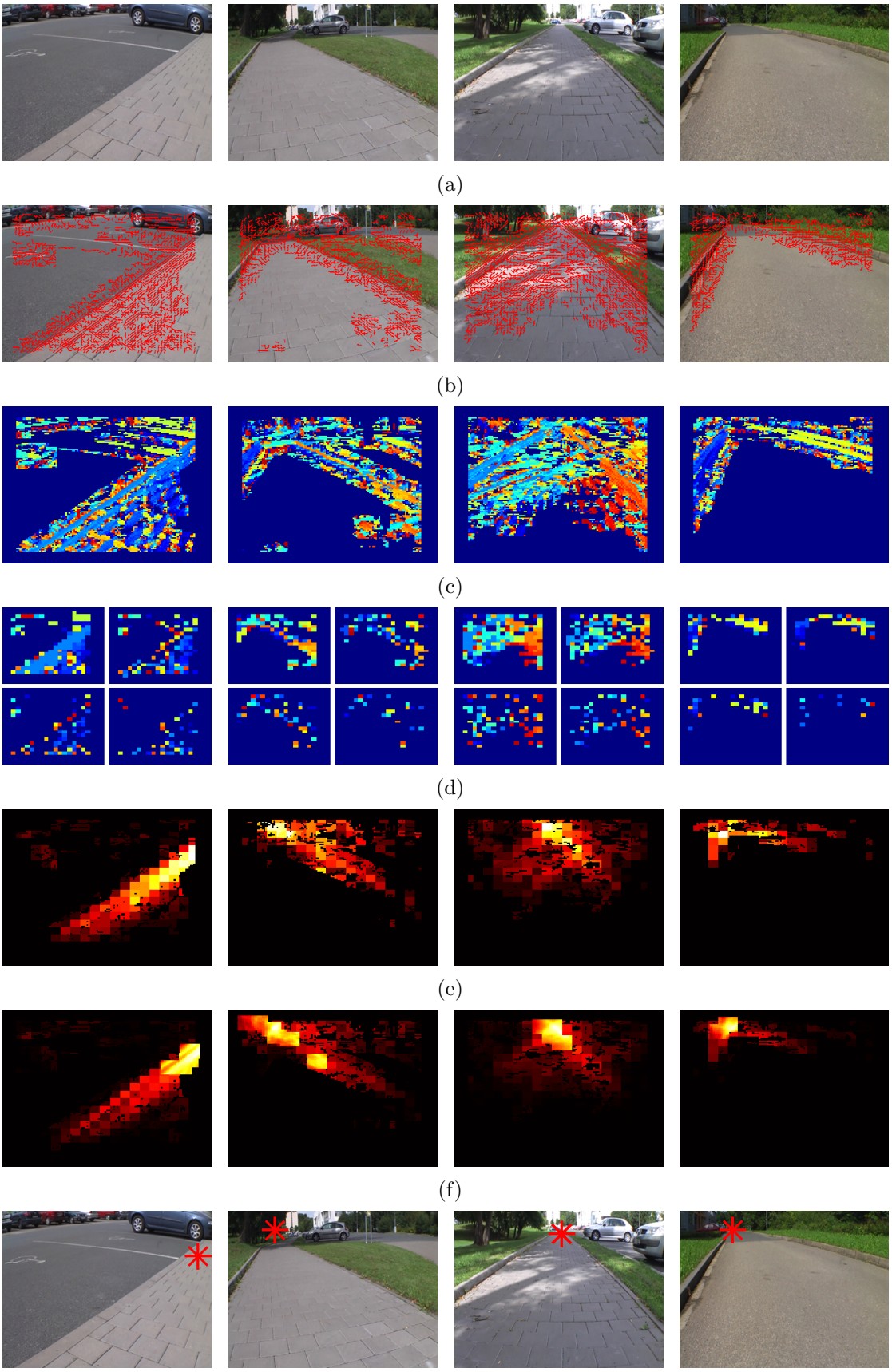


Fig. 5.7: **Vanishing point estimation**: examples – suburban environment – inputs (a), dominant orientations (b,c), superpixels (d), coarse-to-fine voting (e,f), outputs (g).

### 5.1.2 Gaussian Mixture Model

As we have already mentioned in the previous part, the dataset used for vanishing point estimation evaluation consist from still images, i.e. it makes no sense to use this dataset for testing of the road extraction. Moreover, the test sequences of previously published papers [17, 18] are not freely available. Hence, the road extraction algorithm was tested on a number of different sequences, which consist from more than 10 000 images captured in 5Hz by Orpheus-AC around the BUT campus.

By comparison with state-of-the-art methods [17, 18], our training area is defined without any estimation of a 3D depth map. Thus, we are able to distinguish e.g. pavements and other areas like grass, etc. without any high borders. Consequently, our method does not use the whole ground plane, however we are able to select a drivable path with higher precision. On the other hand, due to the HAC we can remove outliers (obstacles, color noise, ...) which differ in either color or height. Sliding of a training area is useful when the robot is close enough to the borders of a path to avoid learning of non-road colors.

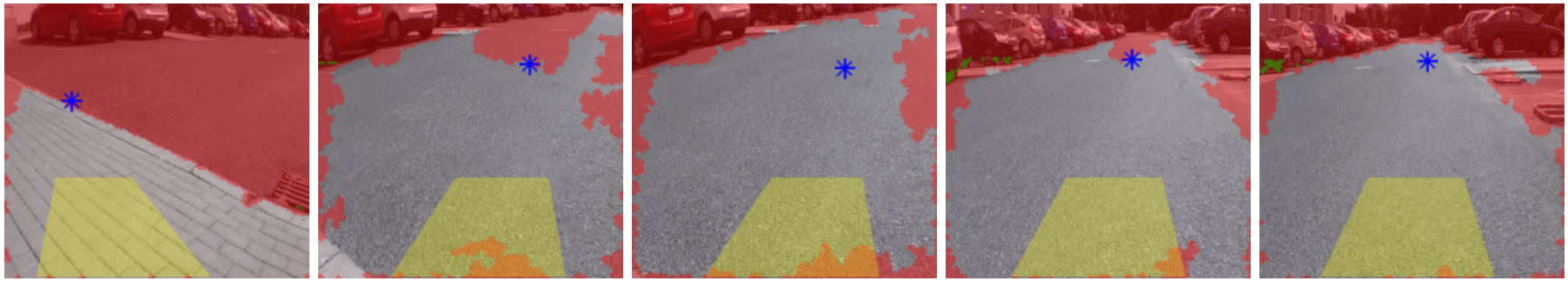
An anti-windup and decay factor are complementary coefficients dealing with better management of the previously learned models in the history archive (see Fig. 5.8). The importance of a sliding training area is shown on Fig. 5.9.

The input images are subsampled to $(I_w, I_h) = (128, 128)$ which is the best trade-off between computational complexity and accuracy. All results were obtained with the following parameters: $n_l = 15$, $h = \frac{4}{5}I_h$, $T_{outliers} = 15\%$, $d_{similar} = 1$, $H_T = 0.2$, $B_T = 30$, $T_{shadow} = 0.1$, $T_{highlight} = 0.85$, $f_{shadow} = 0.2$, $n_{frames_1} = 5$, $n_{frames_2} = 200$ and $c_1c_2c_3$ color space.

#### Examples

The following pages show examples of anti-windup and decay factor (Fig. 5.8), sliding training area (Fig. 5.9) and results obtained on multiple sequences captured on BUT campus (Fig. 5.10). Unfortunately, we do not have any ground-truth data so we do not present any quantitative or qualitative results. Instead, video results[2] are available on enclosed DVD (see Appendix B).

[2]Videos can be also found at `http://www.miksik.co.uk`



(a)

(b)

Fig. 5.8: **Anti-windup**: comparison of road extraction with properly set anti-windup and decay factor (a) and processing without these factors (b). In both cases (even if texture segmentation fails), it is still possible to successfully navigate the robot because the vanishing point can be used.

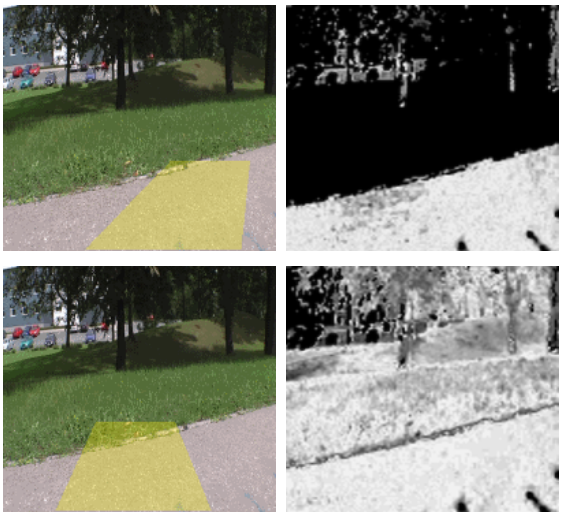


Fig. 5.9: **Sliding training area**: comparison of drivability maps produced with a sliding training area (top row) and fixed training area (bottom row).

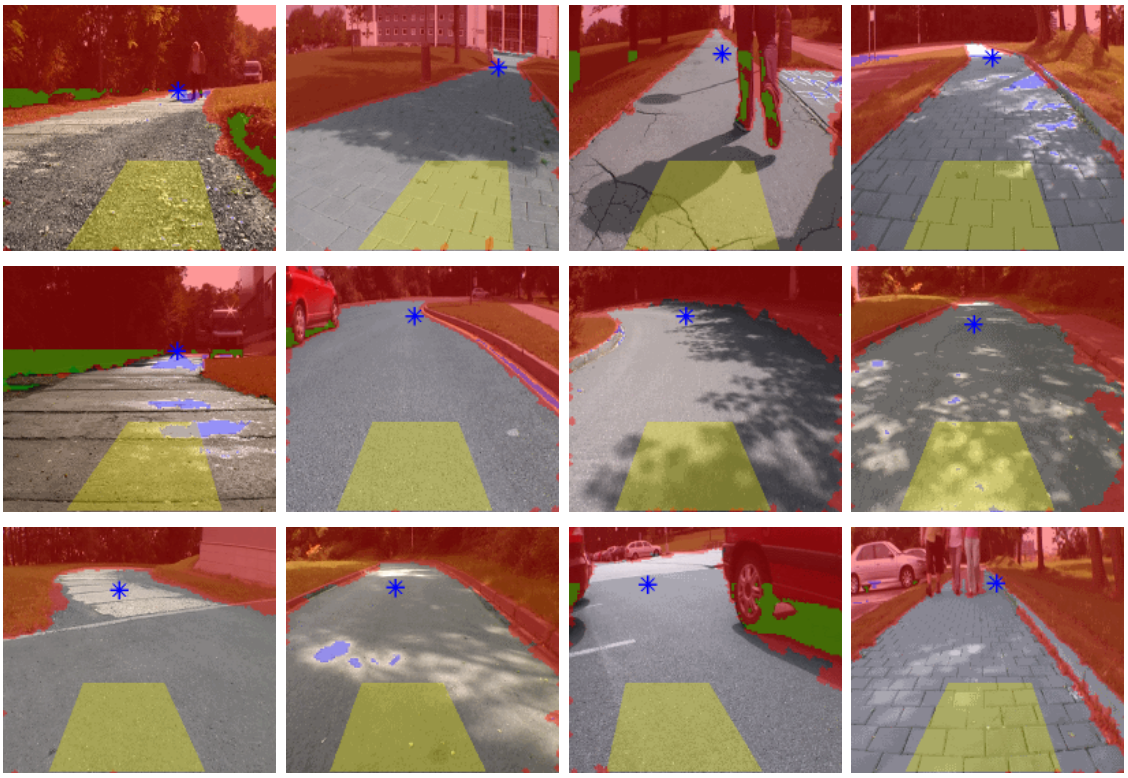


Fig. 5.10: **Fusion of frequency based vanishing point estimation and probabilistically based texture segmentation**: performance against various road types, illumination and obstacles. The blue star is the estimated vanishing point, the yellow trapezoid is a training area, the blue area denotes highlight preprocessor, and the green is the shadow preprocessor.

### 5.1.3 Summary

The proposed algorithm performs well in various environments consisting of different types of pavements, ill-structured rural and (sub)urban roads. The algorithm is robust enough to work in different illumination conditions, including dark cast shadows and overexposed highlights. Shadows and white preprocessors label image areas that do not contain enough information about color. Such information can be used by a higher AI system. Moreover, we are able to specify the adaptivity speed and quantity of models stored in a history archive. Finally, due to the fusion of frequency and probabilistically based approaches, the algorithm is robust against sudden changes of road surfaces.

Although the original vanishing point estimation algorithm was developed for desert roads [66], it works well even in a (sub)urban environment, however it is necessary to use wide lenses to ensure that enough of both road borders will be in the image when there are no significant dominant orientations in the texture. We have optimized computational complexity of a vanishing point estimation algorithm with integral image trick based filtering and coarse-to-fine voting scheme. Such optimization makes the voting scheme more than $40\times$ faster than Kong et al. [39, 38] and $50\times$ faster than Rasmussen [66], while the precision is only $3-5\%$ worse than Kong et al. approach. Moreover, the proposed method is not hardware dependent and might be significantly faster with parallel processing.

Due to the novel fusion of a frequency based vanishing point estimation and a probabilistically based texture segmentation, it can be used even in cases when the road borders are not high, which is the limitation of previous approaches. Dynamic properties can be controlled by complementary anti-windup and decay factors. Besides, a fusion of two different approaches leads to better robustness because even if one of them fails, it is still possible to successfully navigate the robot.

The biggest advantage of the proposed algorithm is its adaptivity – it is not necessary to train the model during the learning phase, etc., which means that the results of the carefully learned models would be better. On the other hand our approach works well in all environments without any significant limitations. The main drawbacks come from the fact that we use only color features for road extraction, i.e. that vanishing point algorithm might fail if the number of dominant orientations estimated on horizontal objects is much higher than the number of ground-plane patches; road extraction might fail if there is no significant color difference between the road/non-road regions.

## 5.2 Spatio-temporal Consistency

The key part of the spatio-temporal filter is the proposed similarity metric. Hence we start with its properties – we show the learned weights and compare the similarity and Euclidean distances on the same features. Then we aim on evaluation of a spatio-temporal filter on the widely-used CamVid database.

### 5.2.1 Similarity Metric

All the results presented in this chapter were obtained with the same metric. This metric was trained on 631 matched keypoints from the Kermit sequence. We generate 1402 positive and negative examples (1402+1402 = 2804). The feature space consists of 141 dimensions (LBP, texture, color) as it is described in sec. 4.5.3. The learning rate $\eta$ was initialized with 0.001 and regulerizer was set to 0.001. At the end of the learning, there were 161 positive and 150 negative (161 + 150 = 311) violated margins.

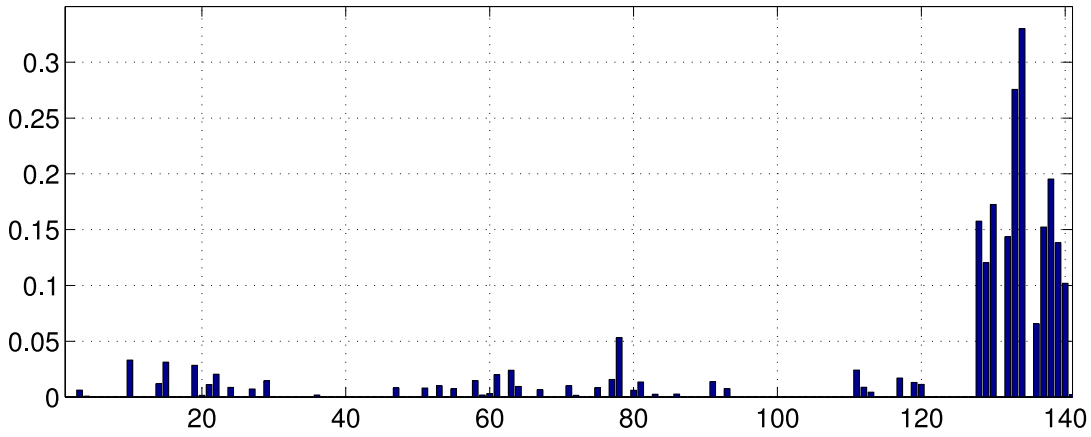
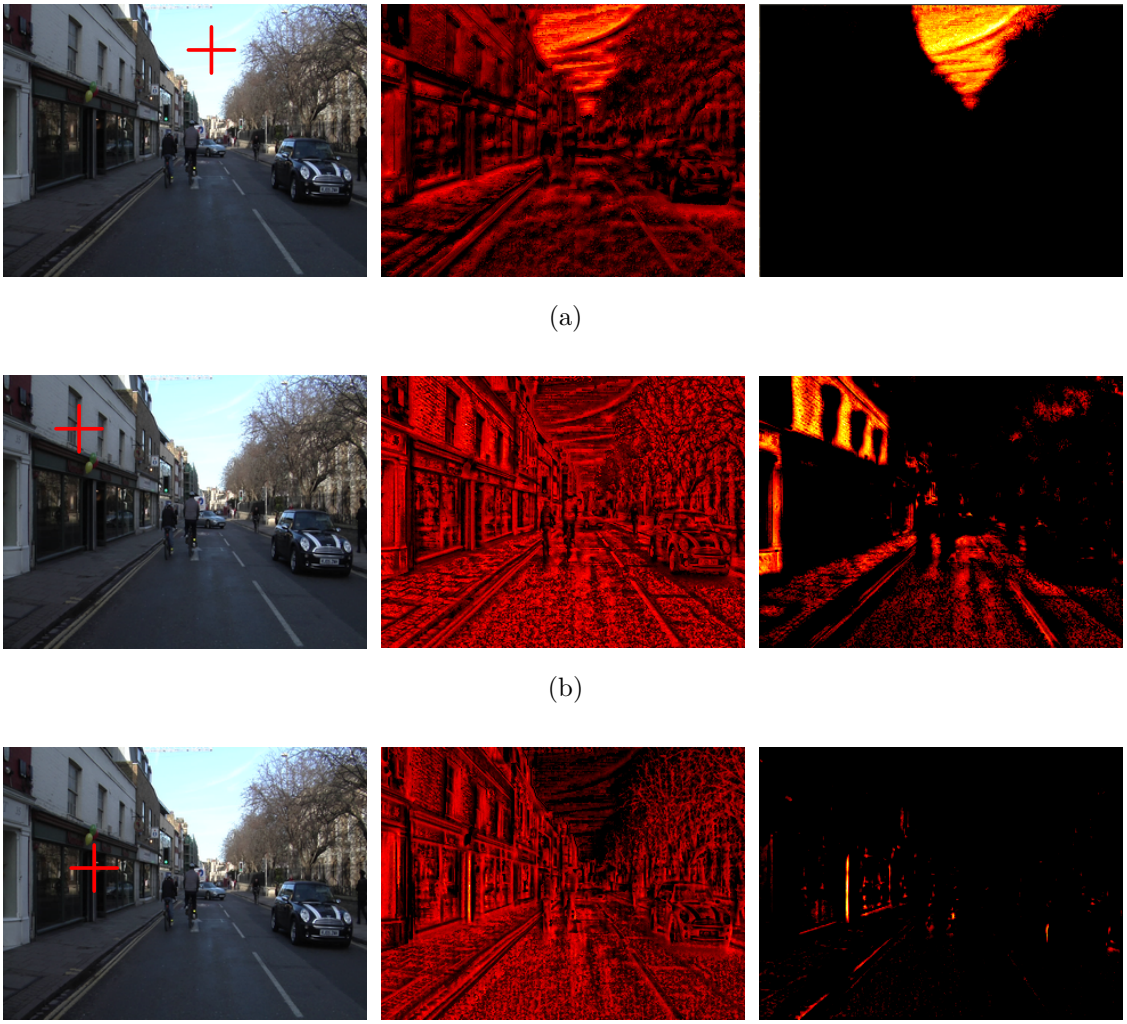



Fig. 5.11: **Similarity distance**: weights – bins [1 : 121] corresponds to LBPs, bins [122 : 138] to texture features and [139 : 141] to color.

Next, we give a few examples of similarity distance compared to Euclidean. Both distances are measured in the same features space (141 dims) and against the same reference pixels. It is obvious, that the similarity metric is much more discriminative than Euclidean distance – the pixels belonging to the different class have larger distances measured with similarity metric to the reference pixel.



(a)

(b)

(c)

Fig. 5.12: **Similarity metric – examples**: first column are input images with reference pixels, second column are Euclidean distances computed on the same feature space as results with similarity metric (last column). The results are shown in inverted "heat" colormap (the brighter the closer), distances measured with similarity metric are truncated to $[0, 2.5]$, with Euclidean distance to $[0, 20]$ for viewing purpose.

### 5.2.2 Spatio-temporal consistency

#### Dataset

Spatio-temporal filter is evaluated on a recently proposed CamVid dataset [10] with high definition ground-truth labels. The CamVid consists of over 10 minutes footage divided into four sequences captured at 30Hz: three sequences were shoot at daylight, the last one was captured at dusk. The sequences were captured mostly at urban and residential environments. The corresponding ground truth labels are at 1Hz and 15Hz for one of the sequences.

The ground-truth labels consist of 32 categories and a small amount of "void" pixels. We follow the other authors [54], [76], [42] and merged them into a subset of 11 categories: Building, Tree, Sky, Car, Sign-Symbol, Road, Pedestrian, Fence, Column-Pole, Sidewalk, and Bicyclist (see Tab. 5.1). Total there are 701 ground-truth images. The database is described more in detail in [10].

We have used Oakland sequence with 6 classes (sky, tree, road, grass, building, foreground) as well, however no ground-truth labels were available for this sequence, so the results are presented as images and video on the enclosed DVD.

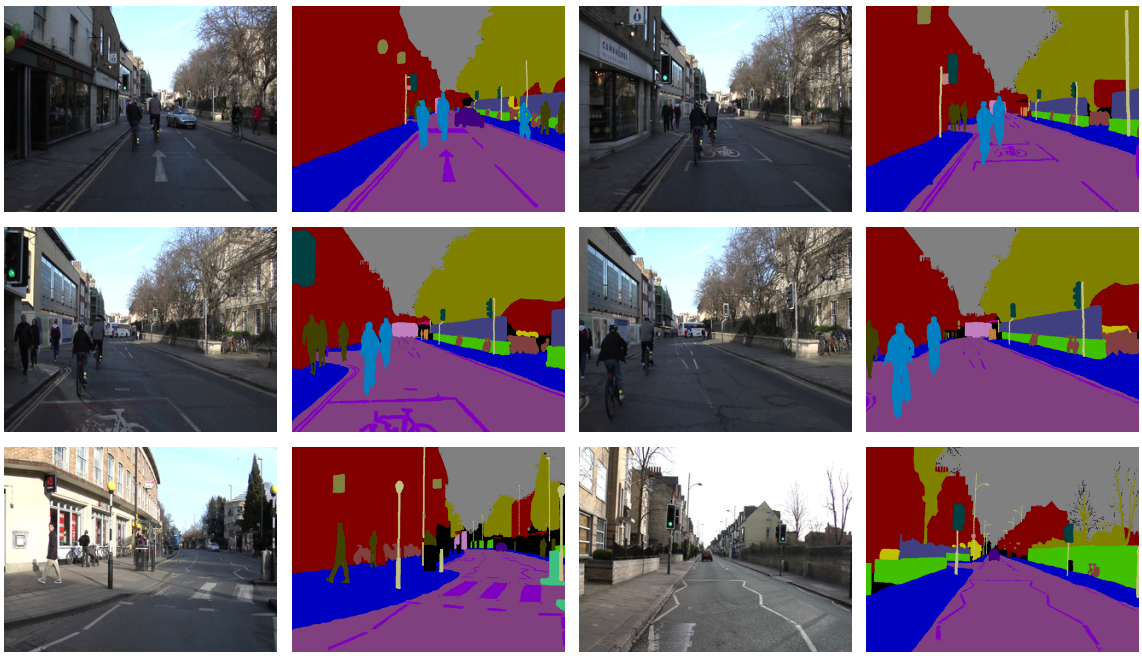


Fig. 5.13: **CamVid dataset – ground-truth**: input images and corresponding ground-truth labels (32 classes, original colormap).

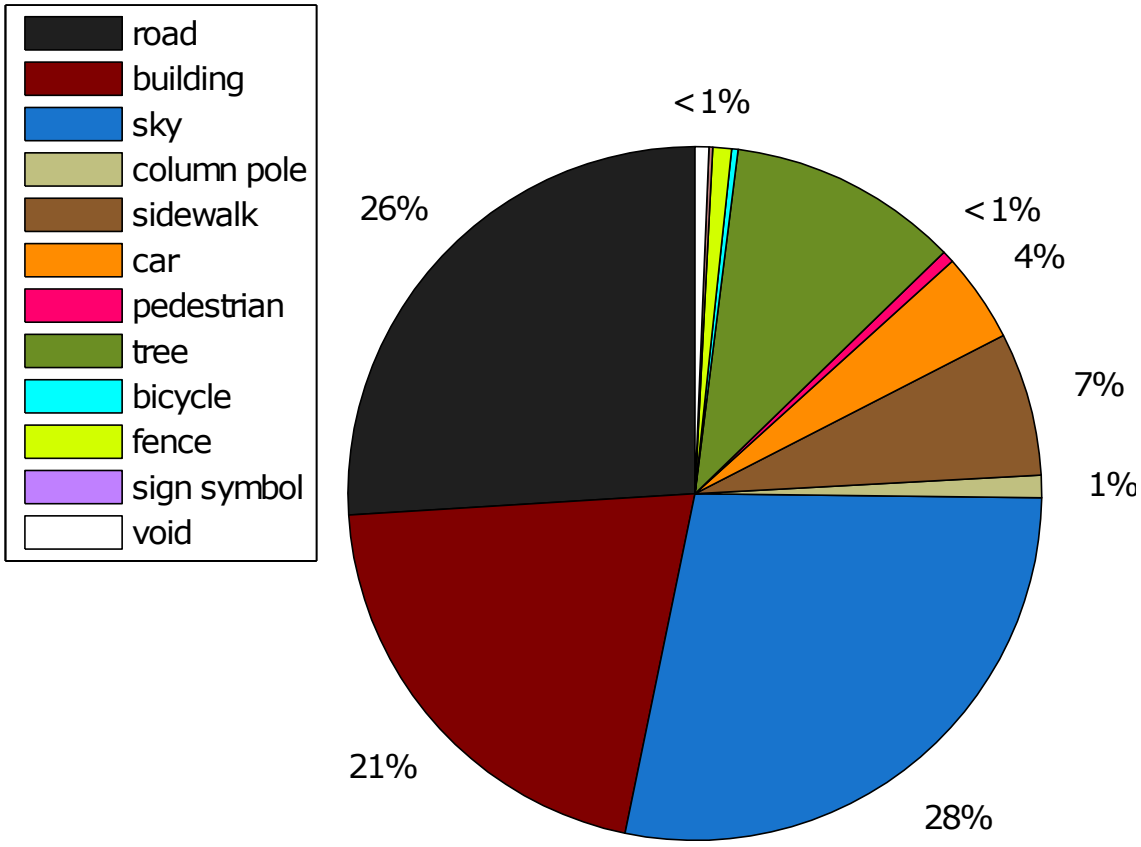



Fig. 5.14: **CamVid – classes.**

Tab. 5.1: **Labeled data**: 32 categories were merged to 11 + void

| **category** | **merged classes** | | | | | % |
|---|---|---|---|---|---|---|
| Road | LaneMkg | RoadShoulder | ParkingBlock | LaneMkg | | 25.98 |
| Building | Archway | Bridge | Tunnel | Misc | Wall | 20.79 |
| Sky | | | | | | 28.04 |
| Column_pole | | | | | | 1.04 |
| Sidewalk | | | | | | 6.69 |
| Car | SUVPickupTruck | Truck_Bus | OtherMoving | Train | CartLuggagePram | 4.15 |
| Pedestrian | Child | Animal | | | | 0.56 |
| Tree | VegetationMisc | | | | | 10.76 |
| Bicycle | MotorcycleScooter | | | | | 0.3 |
| Fence | | | | | | 0.87 |
| Sign_symbol | TrafficLight | TrafficCone | | | | 0.17 |
| *void* | | | | | | 0.65 |

### Experimental setup

Since we need as high density (in time) of ground-truth labels as possible to capture all the flickering effects, we decided to split the images in training (for SHIM) and test data in a following way: sequence with ground-truth labels at 1Hz and 15Hz are used as testing data (148 images), the rest of ground-truths are the training data (396 images). This setup slightly differs from the most widely-used, however we aim on evaluating of spatio-temporal consistency across the video stream and not on pixel-wise labelling of single still images. Hence, it is important to compare the performance against the output of SHIM and not against other methods (MRF, CRF, ...).

All the results were obtained with $\sigma = 0.3$, $c = 0.25$ and the neighborhood consisted of $(5 \times 5)$ pixels.

### Efficiency

First, we give a short comment on efficiency of our algorithm. On average it takes 8.75 seconds per frame. The biggest portion of time is consumed by large displacement optical flow estimation. The good news is that although the parallelization of variational solver is not straightforward, the GPU implementation exists and is up to 80× faster, so the optical flow estimation can be performed in real time (approx. 15 frames per second).

The smoothing filter takes on average 3.42 seconds, however the implementation is very naive and unoptimized. Moreover, the parallelization of our algorithm is very straightforward. Hence, we can conclude that the parallel version of our algorithm can be performed in real-time.

Tab. 5.2: **Efficiency**: averaged run-times

| **algorithm** | **time [s.]** |
|---|---|
| optical flow | 5.33 |
| smoothing | 3.42 |
| **sum** | 8.75 |

### Performance evaluation criteria

It is necessary to consider the fact that our method cannot improve the labelling if the predictions produced by SHIM are constantly wrong. Hence, we use only those

pixels for performance evaluation, which labels obtained by SHIM and by smoothing filter differs. Consequently, we capture only the relative difference against the original predictions.

The results are given in a standard confusion matrix normalized in rows, where the predicted data corresponds to the columns. In addition to that, summarized $F_1$ scores are provided in Tab. 5.3. The $F_1$ score can be interpreted as a weighted average of the precision and recall, where and $F_1$ score reaches its best value at 1 and worst score at 0. The $F_1$ score is defined as

$$\begin{aligned} F_1 &= 2 \cdot \frac{precision \cdot recall}{precision + recall}, \\ s.t. \qquad precision &= \frac{t_p}{t_p + f_n}, \qquad recall = \frac{t_p + t_n}{t_p + t_n + f_p + f_n}, \end{aligned} \tag{5.1}$$

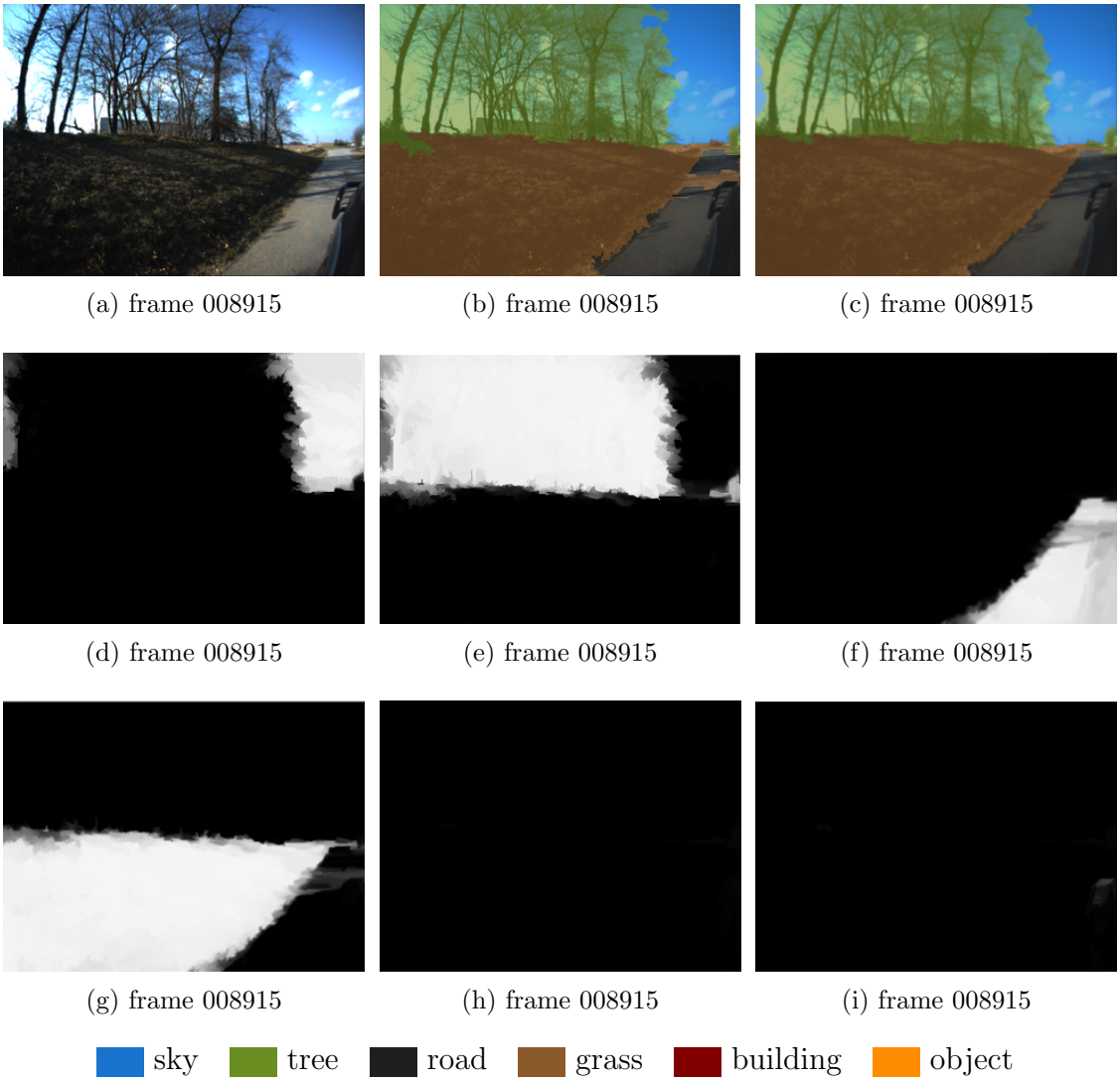
where $t_p$ and $f_p$ are true and false positives, $t_n$ and $f_n$ are true and false negatives, respectively. The total accuracy is defined as

$$accuracy = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \tag{5.2}$$

**Results**

We have used a test sequence which consists of 2000 CamVid frames. The ground-truth labels were available at both, 15Hz (101 frames) and 1Hz (47 frames). The results are summarized in confusion matrices and a table with $F_1$ score and total accuracy. Fig. 5.17 a) shows a confusion matrix for original output, b) is a confusion matrix for proposed method. It is obvious, that the proposed method outperforms the original predictions, especially in the case of larger classes that have higher confidence score. On the other hand, original predictions provides a better performance for small (in term of amount of data) classes like bicyclist or fence (Tab. 5.3). As we have discussed in the previous parts, this is absolutely expected behavior, since it is impossible to distinguish highly unstable predictions from some random flicker.

In addition to that, it is obvious from the confusion matrices, that the more stable results are obtained for classes with higher scores – it means, that the stability of the output might be better, however the precision is worse since the measurement is biased by the quality of original predictions. To clarify that, we give a toy example: consider the periodical flicker with 3 wrong predictions (W) and 1 correct (C), i.e. WWWC and ground truth labels available for each second frame. Then, the original predictions would have much higher score since one frame would be labeled correctly and one incorrectly, although our results are much more stable.




(a) frame 008915 (b) frame 008915 (c) frame 008915

(d) frame 008915 (e) frame 008915 (f) frame 008915

(g) frame 008915 (h) frame 008915 (i) frame 008915

Fig. 5.15: **Oakland dataset – probability maps**: (a) input image, (b) raw labels, (c) smooth labels, probability maps for (d) sky, (e) tree, (f) road, (g) grass, (h) building and (i) object.

Despite the fact that performance evaluation penalizes more stable results, the total accuracy of our method is approximately 47% while the original predictions have accuracy of only 29%. Once again, all the values are measured only on the pixels, that are labeled differently by these two methods. Fig. 5.18 shows typical output of our method compared with original predictions [3].

[3]Videos can be also found at `http://www.miksik.co.uk`

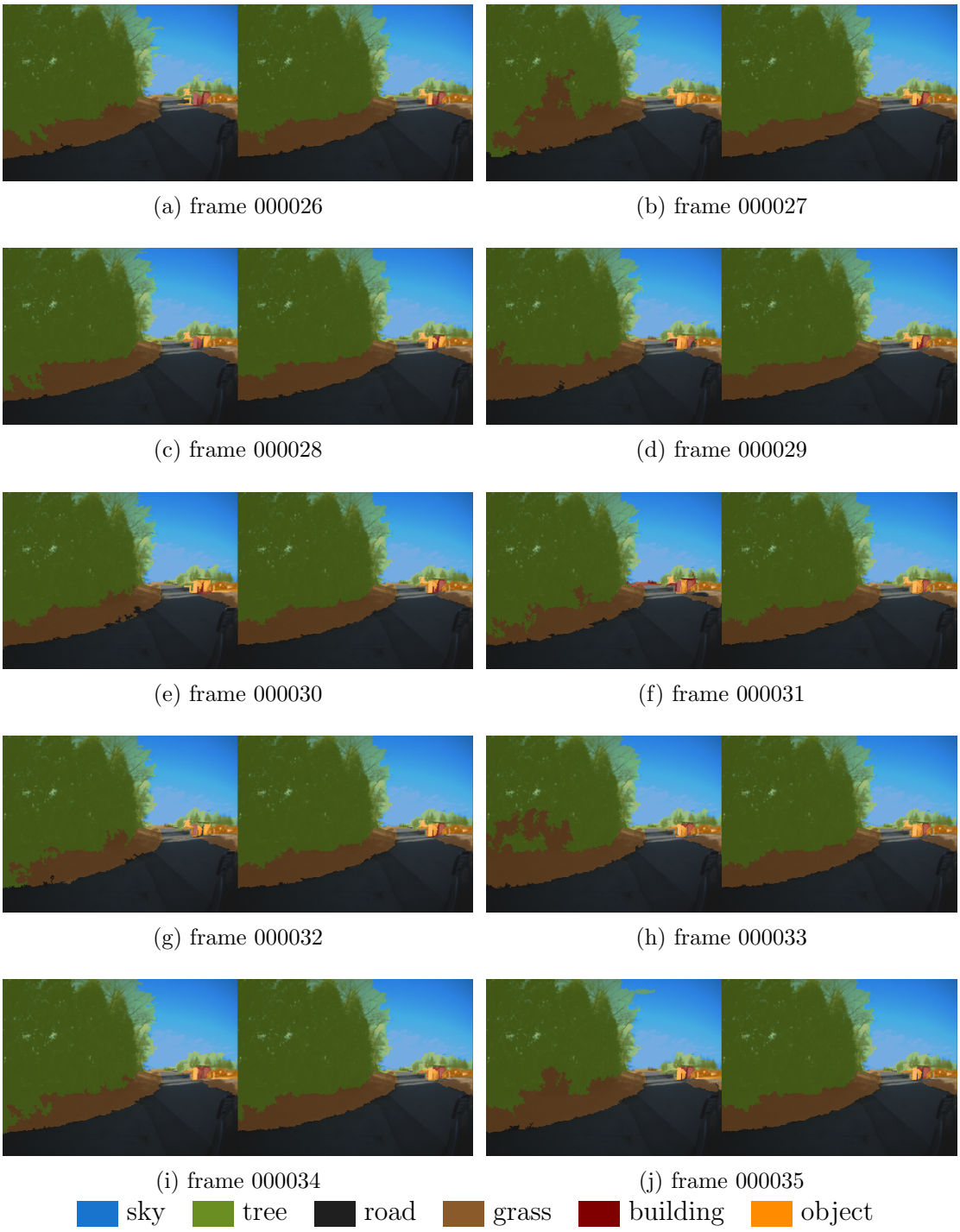
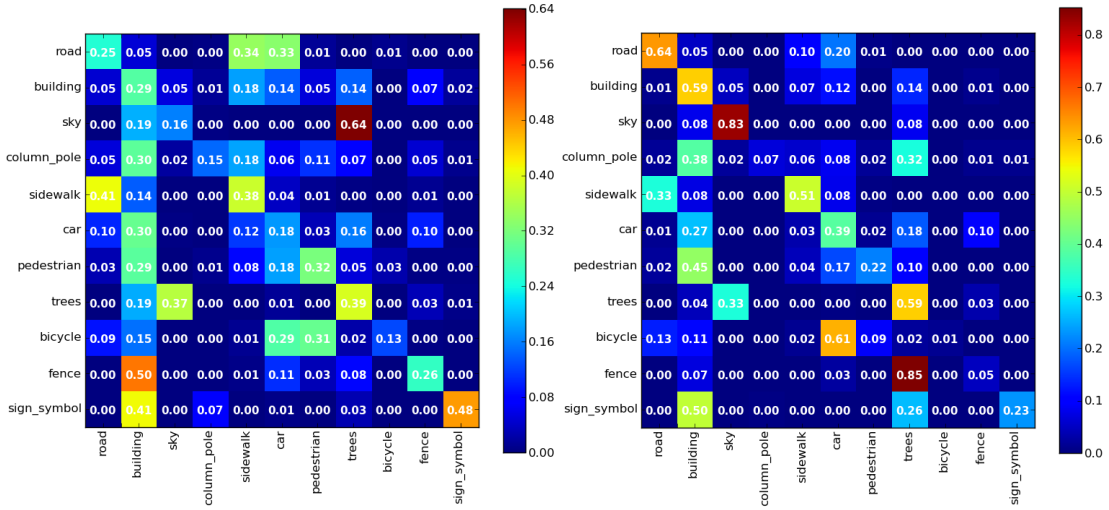



Fig. 5.16: **Oakland dataset – results**: original labels (left part of each image pair), proposed (right part of each image pair).

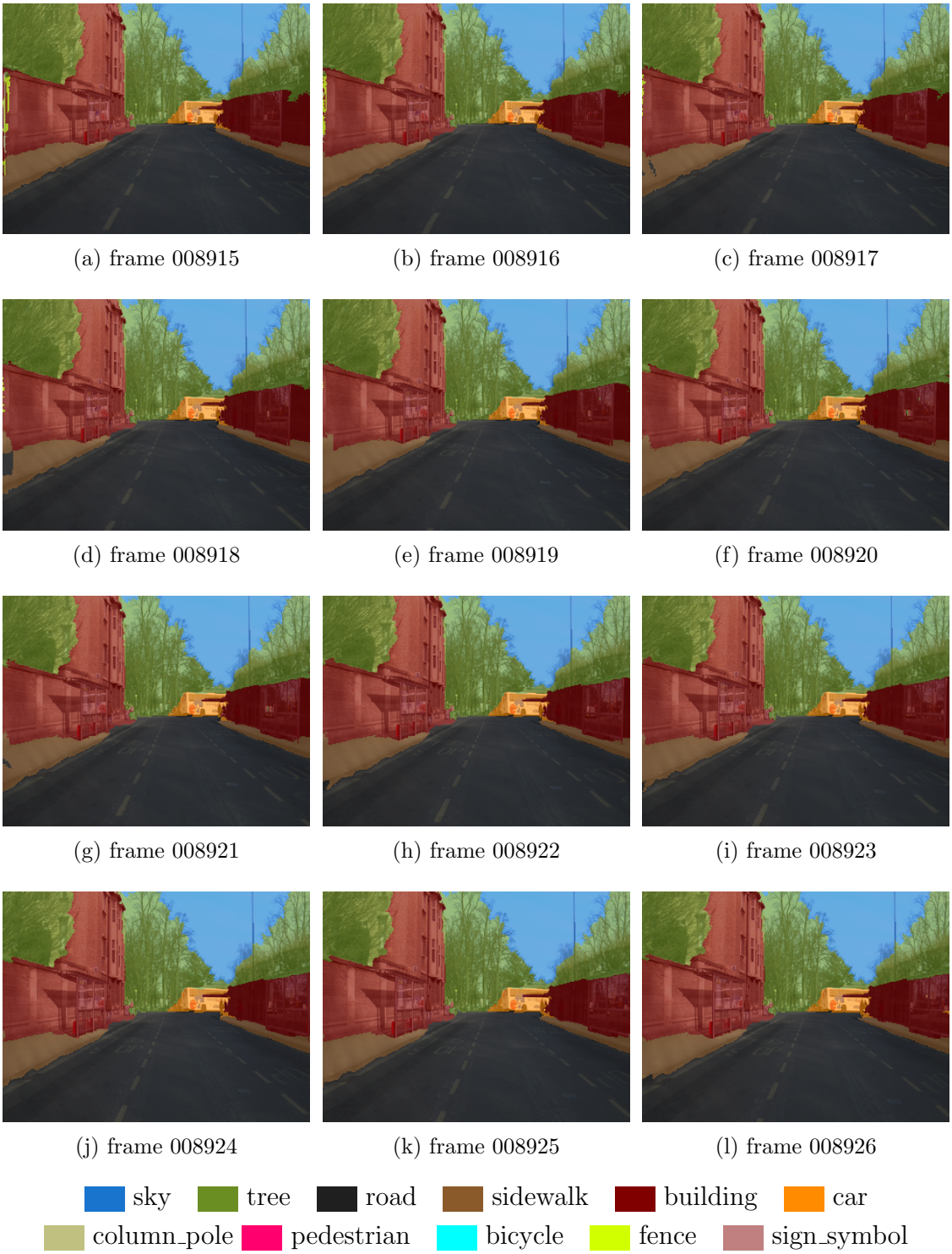



(a) (b)

Fig. 5.17: **CamVid – confusion matrices**: (a) original, (b) proposed.

Tab. 5.3: **CamVid – F1 scores and total accuracy**

| class | original | proposed |
|---|---|---|
| road | 0.2304 | **0.5466** |
| building | 0.3091 | **0.5920** |
| sky | 0.1271 | **0.5375** |
| column_pole | **0.2113** | 0.1227 |
| sidewalk | 0.4305 | **0.6189** |
| car | 0.1092 | **0.2186** |
| pedestrian | 0.2642 | **0.2903** |
| trees | 0.3023 | **0.3588** |
| bicycle | **0.2154** | 0.0283 |
| fence | **0.2693** | 0.0709 |
| sign_symbol | **0.5201** | 0.3574 |
| **total accuracy** | 0.2929 | **0.4728** |



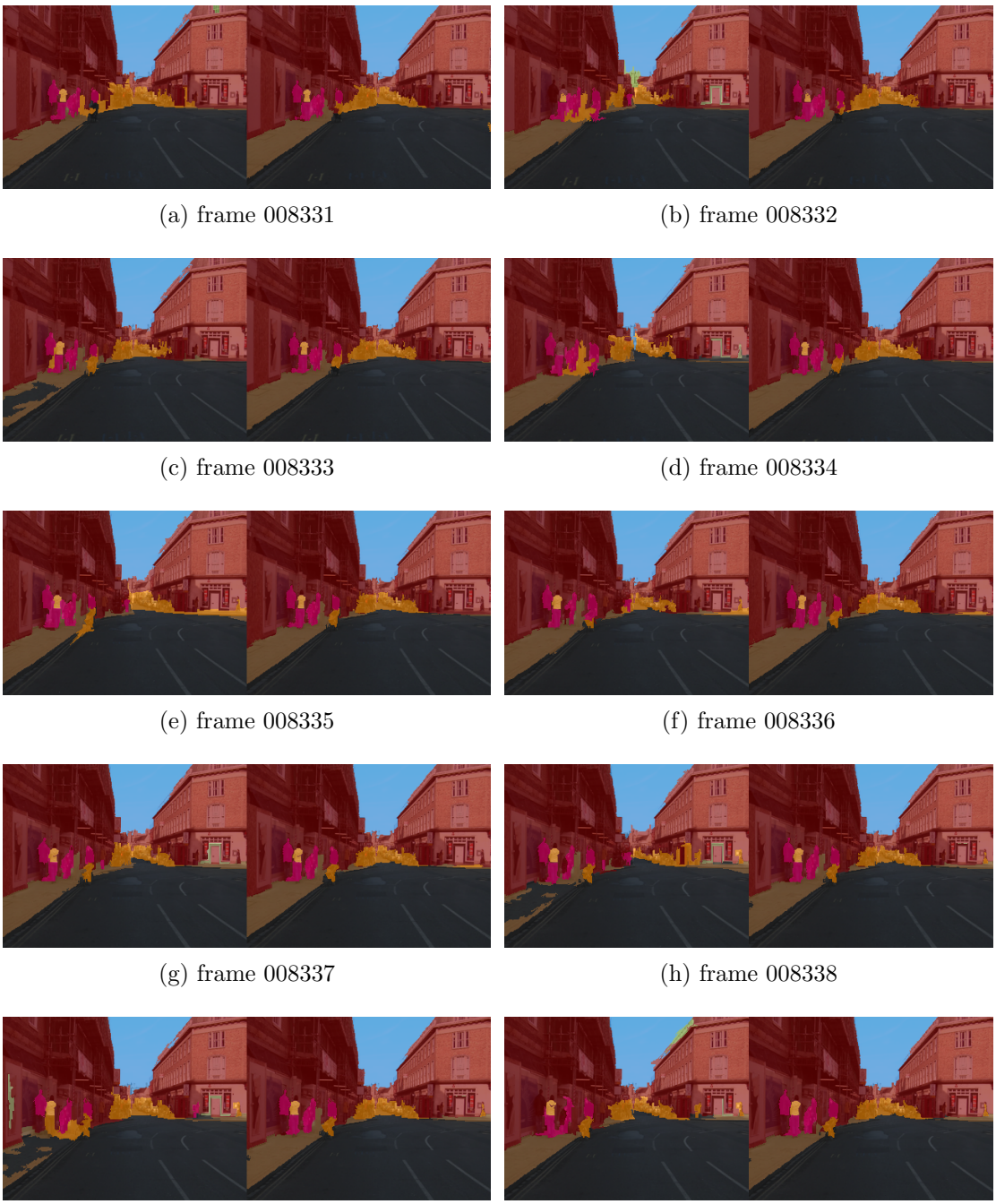

(a) frame 008915 (b) frame 008916 (c) frame 008917

(d) frame 008918 (e) frame 008919 (f) frame 008920

(g) frame 008921 (h) frame 008922 (i) frame 008923

(j) frame 008924 (k) frame 008925 (l) frame 008926

Fig. 5.18: **CamVid – results.**




(a) frame 008331 (b) frame 008332

(c) frame 008333 (d) frame 008334

(e) frame 008335 (f) frame 008336

(g) frame 008337 (h) frame 008338

(i) frame 008339 (j) frame 008340

Fig. 5.19: **CamVid dataset – results**: original labels (left part of each image pair), proposed (right part of each image pair).

### 5.2.3 Summary

The proposed approach works well in both, urban and suburban environments. Although we used the smoothing filter with predictions obtained by stacked hierarchical inference machine, the model itself is not restricted to these predictions and can be combined with any other approach to pixel-wise labelling aiming at both, single still images and dynamic scenes as well. The only assumption we made is that we expect the front/back view camera. The side view cameras can be used as well, however the processing requires additional information since all the objects and stuff usually appears in side view cameras just for a few frames (e.g. 5 frames) and thus, it is difficult to distinguish random flicker on moving objects from correct labels.

We have proposed a similarity metric that is more discriminative than standard Euclidean distance in the same feature space. The proposed metric is learned on LBPs, texture and color features. Hence, it is able to distinguish even the pixels having very similar color appearance (typically caused by some blur, etc.).

We did a number of various experiments and in general, the smoothing filter outperforms raw predictions, however the devil is hidden in detail. Despite the fact that we use the most advanced large displacement optical flow estimation, situations exist when the optical flow estimation fails (especially on small objects).

It is difficult to proof it since the LDOF code is not available, however our suspicions aim on descriptors matching used by LDOF. This is for the following reasons: (1) descriptors matching is the key part dealing with large displacements. (2) if the camera moves straight ahead and the small object is far away from the camera center, the displacements of both, the object and the stuff behind are almost the same and hence, even though the matching fails, the optical flow is estimated correctly, because of the used variational method. However, if the object is closer to the camera, especially during the rotation, the displacements of the object and stuff differs. In such situation, if the matching fails, then the variational approach "somehow" estimates the optical flow, however such flow is obviously incorrect. The main issue is that we are able to detect only the failures caused by occlusions, however not such failures, since both, the forward and backward flows are the same.

Last but not least, we should highlight that our approach is easy-to-implement since it does not need any "hidden" tricks, etc.

# 6 CONCLUSION

This thesis deals with dynamic scene understanding for mobile robot navigation. The goal was to assign a class label to each pixel in a frame. This thesis has two main parts that can be combined together and can be used for various applications.

The first part of the thesis proposed a self-supervised learning algorithm for road extraction, that can be used with robots such as Orpheus-AC, e.g. in the case of signal loss, etc. The algorithm estimates the vanishing point of the road which defines the training area for self-supervised learning of the Gaussian mixture model for road extraction.

We have optimized computational complexity of a vanishing point estimation algorithm by approximation of Gabor wavelets with Haar-like binary box functions, which enables fast filtering via integral image trick. The tightest bottleneck of a vanishing point estimation algorithm was voting. We have proposed a novel, coarse-to-fine scheme. The proposed algorithm is up to $40\times$ faster than Kong. et al. [39], [38] and $50\times$ faster than Rasmussen [66], while the precision is only $3-5\%$ worse than Kong et al. approach. It is important, that the method is not hardware dependent and would be significantly faster with parallel processing.

Due to the novel fusion of a frequency based vanishing point estimation and probabilistically based texture segmentation, it can be used even in cases when the road borders are no height, which is the limitation of previous approaches. Dynamic properties can be controlled by complementary anti-windup and decay factors. Besides, a fusion of two different approaches leads to better robustness because if one of them fails, it is still possible to successfully navigate the robot. The proposed approach works well on both unstructured and (semi)structured roads, with various types of surfaces and dynamically changing light conditions including dark cast shadows and overexposed highlights.

The second part deals with spatio-temporal consistency of pixel-wise labeling algorithms, such as stacked hierarchical inference machine [59]. Such methods are useful for more mature systems, such as self-driving cars since they provide more information about the perceived environments. The key part of proposed smoothing filter is a new similarity metric, which is more discriminative than the standard Euclidean distance and can be used for various computer vision tasks. The smoothing filter first estimates optical flow to define a local neighborhood. This neighborhood is used for recursive averaging, based on the similarity metric. The total accuracy of

proposed method measured on pixels with inconsistent labels between the raw and smooth predictions is almost 18% higher than original predictions. Although, we have used SHIM, the algorithm can be combined with any other system for structured predictions (MRF/CRF, ...). The proposed smoothing filter represents a first step towards full inference.

Since the communication of ideas and research is important, several papers have been already published, and others are coming soon.

# BIBLIOGRAPHY


[1] Andrle, M.; Rebollo-Neira, L.: A swapping-based refinement of orthogonal matching pursuit strategies. *Signal Process.*, volume 86, March 2006: pages 480–495, ISSN 0165-1684, doi:10.1016/j.sigpro.2005.05.034.

[2] Barinova, O.; Konushin, V.; Yakubenko, A.; et al.: Fast Automatic Single-View 3-d Reconstruction of Urban Scenes. In *Proceedings of the 10th European Conference on Computer Vision: Part II*, ECCV '08, Berlin, Heidelberg: Springer-Verlag, 2008, ISBN 978-3-540-88685-3, pages 100–113.

[3] Bay, H.; Ess, A.; Tuytelaars, T.; et al.: Speeded-Up Robust Features (SURF). *CVIU*, volume 110, nr. 3, 2008.

[4] Berger, K.; Lipski, C.; Linz, C.; et al.: The area processing unit of caroline - finding the way through darpa's urban challenge. RobVis. 2008.

[5] Bertozzi, M.; Broggi, A.; Fascioli, A.; et al.: Stereo Vision-based Vehicle Detection. In *in IEEE Intelligent Vehicles Symposium*, 2000, pages 39–44.

[6] Borenstein, J.; Everett, H. R.; Feng, L.: *Where am I? Systems and Methods for Mobile Robot Positioning*. March 1996.

[7] Broggi, A.; Bertozzi, M.; Fascioli, A.: ARGO and the MilleMiglia in Automatico Tour. *IEEE Intelligent Systems*, volume 14, nr. 1, 1999: pages 55–64, ISSN 1541-1672, doi:http://dx.doi.org/10.1109/5254.747906.

[8] Broggi, A.; Bombini, L.; Cattani, S.; et al.: Sensing requirements for a 13,000 km intercontinental autonomous drive. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, june 2010, ISSN 1931-0587, pages 500 –505, doi:10.1109/IVS.2010.5548026.

[9] Broggi, A.; Fascioli, A.; Member, A.; et al.: Artificial Vision in Extreme Environments for Snowcat Tracks Detection. *IEEE Trans. on Intelligent Transportation Systems*, volume 3, 2002: pages 162–172.

[10] Brostow, G. J.; Fauqueur, J.; Cipolla, R.: Semantic object classes in video: A high-definition ground truth database. *Pattern Recogn. Lett.*, volume 30, nr. 2, January 2009: pages 88–97, ISSN 0167-8655.

[11] Brostow, G. J.; Shotton, J.; Fauqueur, J.; et al.: Segmentation and Recognition using Structure from Motion Point Clouds. In *European Conference on Computer Vision (ECCV)*, 2008, pages 1–14.

[12] Brox, T.; Malik, J.: Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence*, volume 33, nr. 3, March 2011: pages 500–13, ISSN 1939-3539, doi:10.1109/TPAMI.2010.143.
URL `http://www.ncbi.nlm.nih.gov/pubmed/20714020`

[13] Calonder, M.; Lepetit, V.; Strecha, C.; et al.: BRIEF: Binary Robust Independent Elementary Features. In *ECCV*, 2010.

[14] Crisman, J.; Thorpe, C.: UNSCARF, A Color Vision System for the Detection of Unstructured Roads. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, April 1991, pages 2496 – 2501.

[15] Crisman, J.; Thorpe, C.: SCARF: A Color Vision System that Tracks Roads and Intersections. *IEEE Trans. on Robotics and Automation*, volume 9, nr. 1, February 1993: pages 49 – 58.

[16] Crow, F. C.: Summed-area tables for texture mapping. *SIGGRAPH Comput. Graph.*, volume 18, nr. 3, January 1984: pages 207–212, ISSN 0097-8930.

[17] Dahlkamp, H.; Kaehler, A.; Stavens, D.; et al.: Self-supervised Monocular Road Detection in Desert Terrain. In Sukhatme et al. [78].

[18] Dong-Si, T.-C.; Guo, D.; Yan, C. H.; et al.: Robust extraction of shady roads for vision-based UGV navigation. In *IROS*, IEEE, 2008, pages 3140–3145.

[19] Ess, A.; Müller, T.; Grabner, H.; et al.: Segmentation-based urban traffic scene understanding. In *British machine vision conference (BMVC)*, 2009, pages 1–11.
URL `http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Segmentation-Based+Urban+Traffic+Scene+Understanding#0`

[20] Felzenszwalb, P. F.; Huttenlocher, D. P.: Efficient graph-based image segmentation. *International Journal of Computer Vision (IJCV)*, 2004: pages 1–26.
URL `http://www.springerlink.com/index/n8110427355x2312.pdf`

[21] Finlayson, G. D.; Schaefer, G.: Hue that is invariant to brightness and gamma. In *BMVC*, edited T. F. Cootes; C. J. Taylor, British Machine Vision Association, 2001, ISBN 1-901725-16-2.

[22] Ghurchian, R.; Hashino, S.: Shadow Compensation in Color Images for Unstructured Road Segmentation. In *MVA*, 2005, ISBN 4-901122-04-5, pages 598–601.

[23] Grudic, G. Z.; Mulligan, J.: Outdoor Path Labeling Using Polynomial Mahalanobis Distance. In Sukhatme et al. [78].

[24] Grundmann, M.; Kwatra, V.; Han, M.; et al.: Efficient hierarchical graph-based video segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2010, pages 2141–2148.
URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5539893`

[25] Gupta, A.; Efros, A. A.; Hebert, M.: Blocks World Revisited: Image Understanding using Qualitative Geometry and Mechanics. In *European Conference on Computer Vision (ECCV)*, September 2010.

[26] Hansen, M.; Sommer, G.: Active Depth Estimation with Gaze and Vergence Control using Gabor filters. In *In In 13th Int. Conf. on Pattern Recognition, Volume A*, 1996, pages 287–291.

[27] Hanson, A.; Riseman, E.: VISIONS: A computer system for interpreting scenes. *Computer Vision Systems*, 1978: pages 303–333.

[28] He, X.; Zemel, R. S.; Carreira-Perpiñán, M. Á.: Multiscale Conditional Random Fields for Image Labeling. In *CVPR (2)*, 2004, pages 695–702.

[29] Hoiem, D.; Efros, A. A.; Hebert, M.: Closing the loop in scene interpretation. *Computer Vision and Pattern Recognition (CVPR)*, 2008.
URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4587587`

[30] Hoiem, D.; Efros, A. a.; Hebert, M.: Putting Objects in Perspective. *International Journal of Computer Vision*, volume 80, nr. 1, April 2008: pages 3–15, ISSN 0920-5691, doi:10.1007/s11263-008-0137-5.
URL `http://www.springerlink.com/index/10.1007/s11263-008-0137-5`

[31] Isard, M.; Blake, A.: CONDENSATION - Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, volume 29, nr. 1, 1998: pages 5–28.

[32] Jones, J. P.; Palmer, L. A.: An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *J Neurophysiol*, volume 58, nr. 6, December 1987: pages 1233–1258, ISSN 0022-3077.

[33] Kalal, Z.; Mikolajczyk, K.; Matas, J.: Forward-Backward Error : Automatic Detection of Tracking Failures. In *International Conference on Pattern Recognition*, 2010, pages 23–26.

[34] Kohli, P.; Kumar, M. P.; Torr, P. H. S.: P3 & Beyond: Solving Energies with Higher Order Cliques. In *CVPR*, IEEE Computer Society, 2007.

[35] Kohli, P.; Ladicky, L.; Torr, P. H. S.: Robust Higher Order Potentials for Enforcing Label Consistency. In *Computer Vision and Pattern Recognition (CVPR)*, volume 01, 2008.

[36] Kohli, P.; Ladicky, L.; Torr, P. H. S.: Robust Higher Order Potentials for Enforcing Label Consistency. *International Journal of Computer Vision*, volume 82, nr. 3, 2009: pages 302–324.

[37] Koller, D.; Friedman, N.: *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[38] Kong, H.; Audibert, J.-Y.; Ponce, J.: General road detection from a single image. *IEEE Transactions on Image Processing*, volume 19, nr. 8, 2009: pages 2211–2220.

[39] Kong, H.; Audibert, J.-Y.; Ponce, J.: Vanishing point detection for road detection. *IEEE Conference on Computer Vision and Pattern Recognition (2008)*, , nr. 3, 2009: pages 96–103.

[40] Kumar, S.; Hebert, M.: Discriminative Random Fields : A Discriminative Framework for Contextual Interaction in Classification. In *International Conference on Computer Vision (ICCV)*, IEEE, 2003.

[41] Kumar, S.; Hebert, M.: A hierarchical field framework for unified context-based classification. In *International Conference on Computer Vision (ICCV)*, IEEE, 2005, ISBN 0-7695-2334-X, pages 1284–1291 Vol. 2, doi:10.1109/ICCV.2005.9. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1544868`

[42] Ladicky, L.; Sturgess, P.; Alahari, K.; et al.: What, Where & How Many? Combining Object Detectors and CRFs. In *Proceedings of European Conference on Computer Vision*, 2010.

[43] Lafferty, J. D.; McCallum, A.; Pereira, F. C. N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, edited C. E. Brodley; A. P. Danyluk, Morgan Kaufmann, 2001, ISBN 1-55860-778-1, pages 282–289.

[44] Lee, T. S.: Image Representation Using 2D Gabor Wavelets. *IEEE Trans. Pattern Analysis and Machine Intelligence*, volume 18, 1996: pages 959–971.

[45] Leung, T.; Malik, J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision (IJCV)*, volume 43, nr. 1, 2001: pages 29–44.

[46] Leutenegger, S.; Chli, M.; Siegwart, R.: BRISK: Binary Robust invariant scalable keypoints. In *ICCV*, 2011.

[47] Lezama, J.; Alahari, K.; Sivic, J.; et al.: Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *Computer Vision and Pattern Recognition (CVPR)*, Ieee, June 2011, ISBN 978-1-4577-0394-2, pages 3369–3376, doi:10.1109/CVPR.2011.6044588.
URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6044588`

[48] Lieb, D.; Lookingbill, A.; Thrun, S.: Adaptive Road Following using Self-Supervised Learning and Reverse Optical Flow. In *Robotics: Science and Systems*, edited S. Thrun; G. S. Sukhatme; S. Schaal, The MIT Press, 2005, ISBN 0-262-70114-6, pages 273–280.

[49] Liu, C.; Yuen, J.; Torralba, A.: SIFT flow: dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, volume 33, nr. 5, May 2011: pages 978–94, ISSN 1939-3539, doi:10.1109/TPAMI.2010.147.
URL `http://www.ncbi.nlm.nih.gov/pubmed/20714019`

[50] Liu, C.; Yuen, J.; Torralba, A.; et al.: SIFT Flow: Dense Correspondence across Different Scenes. 2008.

[51] Lourakis, M. A.; Argyros, A.: SBA: A Software Package for Generic Sparse Bundle Adjustment. *ACM Trans. Math. Software*, volume 36, nr. 1, 2009: pages 1–30, doi:http://doi.acm.org/10.1145/1486525.1486527.

[52] Lowe, D. G.: Object Recognition from Local Scale-Invariant Features. In *ICCV*, 1999, pages 1150–1157.

[53] Ma, B.; Lakshmanan, S.; Hero, A. O.: Simultaneous detection of lane and pavement boundaries using model-based multisensor fusion. *IEEE Transactions on Intelligent Transportation Systems*, 2000.

[54] Micusik, B.; Kosecka, J.: Semantic segmentation of street scenes by superpixel co-occurrence and 3D geometry. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, IEEE, September 2009, ISBN 978-1-4244-4442-7, pages 625–632.

[55] Mikolajczyk, K.; Schmid, C.: A Performance Evaluation of Local Descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, volume 27, nr. 10, 2005: pages 1615–1630, ISSN 0162-8828.

[56] Miksik, O.; Mikolajczyk, K.: Local Detectors and Descriptor for Fast Feature Matching. In *ICPR*, 2012, under review.

[57] Miksik, O.; Petyovsky, P.; Zalud, L.; et al.: Robust Detection of Shady and Highlighted Roads for Monocular Camera Based Navigation of UGV. In *International Conference on Robotics and Automation (ICRA)*, Shanghai, 2011.
URL `http://www.miksik.co.uk`

[58] Miksik, O.; Petyovsky, P.; Zalud, L.; et al.: Robust Detection of Shady and Highlighted Roads for Monocular Camera Based Navigation of UGV. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[59] Munoz, D.; Bagnell, J. A.; Hebert, M.: Stacked Hierarchical Labeling. In *European Conference on Computer Vision (ECCV)*, IEEE, 2010.

[60] Ochs, P.; Brox, T.: Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.
URL `http://lmb.informatik.uni-freiburg.de/Publications/2011/OB11/ochs_iccv11.pdf`

[61] Panoramio. [online], Date of visit to site 24.4.2012.
URL `http://www.panoramio.com`

[62] Payne, B. R.; Belkasim; Owen, S. G.; et al.: *Accelerated 2D Image Processing on GPUs*, volume 3515. January 2005, 256–264 pages, doi:10.1007/11428848_32.

[63] Pomerleau, D.: Neural Network Vision for Robot Driving. In *The Handbook of Brain Theory and Neural Networks*, edited M. Arbib, 1995.

[64] Pomerleau, D.: RALPH: Rapidly Adapting Lateral Position Handler. In *IEEE Symposium on Intelligent Vehicles*, September 1995, pages 506 – 511.

[65] Prince, S. J. D.: *Computer vision : models , learning and inference*. Cambridge University Press, 2012.

[66] Rasmussen, C.: Grouping Dominant Orientations for Ill-Structured Road Following. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2004.

[67] Rasmussen, C.: A Hybrid Vision + Ladar Rural Road Follower. In *IEEE International Conference on Robotics and Automation*, 2006.

[68] Rasmussen, C.; Korah, T.: On-Vehicle and Aerial Texture Analysis for Vision-Based Desert Road Following. In *IEEE International Workshop on Machine Vision for Intelligent Vehicles*, 2005.

[69] Rauskolb, F. W.; Berger, K.; Lipski, C.; et al.: Caroline: An autonomously driving vehicle for urban environments. *J. Field Robot.*, volume 25, nr. 9, 2008: pages 674–724, ISSN 1556-4959, doi:http://dx.doi.org/10.1002/rob.v25:9.

[70] Rebollo-Neira, L.; Lowe, D.: Optimized Orthogonal Matching Pursuit Approach. *IEEE Signal Processing Letters*, volume 9,4, 2002: pages 137–140.

[71] Richter, M.; Petyovsky, P.; Miksik, O.: Adapting Polynomial Mahalanobis Distance for Self-supervised Learning in an Outdoor Environment. In *International Conference on Machine Learning and Applications (ICMLA)*, 2011.

[72] Rosten, E.; Drummond, T.: Machine learning for high-speed corner detection. In *In ECCV*, 2006.

[73] Rublee, E.; Rabaud, V.; Konolige, K.; et al.: ORB: An efficient alternative to SIFT or SURF. In *ICCV*, 2011.

[74] Snavely, N.; Seitz, S. M.; Szeliski, R.: Photo Tourism: Exploring Photo Collections in 3D. In *ACM TRANSACTIONS ON GRAPHICS*, Press, 2006, pages 835–846.

[75] Stanford Racing. [online], Date of visit to site 24.4.2012.
URL `http://cs.stanford.edu/group/roadrunner//old/index.html`

[76] Sturgess, P.; Alahari, K.; Ladicky, L.; et al.: Combining Appearance and Structure from Motion Features for Road Scene Understanding. In *British Machine Vision Conference (BMVC)*, 2009, pages 1–11.

[77] Sturgess, P.; Alahari, K.; Russell, C.; et al.: What , Where & How Many ? Combining Object Detectors and CRFs. In *European Conference on Computer Vision (ECCV)*, 2010.

[78] Sukhatme, G. S.; Schaal, S.; Burgard, W.; et al. (editors): *Robotics: Science and Systems II, August 16-19, 2006. University of Pennsylvania, Philadelphia, Pennsylvania, USA*, The MIT Press, 2007, ISBN 0-262-69348-8.

[79] Sundaram, N.; Brox, T.; Keutzer, K.: Dense Point Trajectories by GPU-accelerated Large Displacement Optical Flow. Technical report UCB/EECS-2010-104, EECS Department, University of California, Berkeley, Jul 2010.
URL http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-104.html

[80] Sundaram, N.; Brox, T.; Krutzer, K.: Dense point trajectories by GPU-accelerated large displacement optical flow. In *European Conference on Computer Vision (ECCV)*, 2010.
URL http://www.springerlink.com/index/DWPX5R1N6K463432.pdfhttp://www.springerlink.com/index/dwpx5r1n6k463432.pdf

[81] Tang, F.; Crabb, R.; Tao, H.: Representing images using nonorthogonal Haar-like bases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 29, nr. 12, 2007: pages 2120–2134.

[82] Tang, F.; Tao, H.: Non-orthogonal Binary Expansion of Gabor Filters with Applications in Object Tracking. *2007 IEEE Workshop on Motion and Video Computing WMVC07*, 2007: pages 24–24.

[83] DARPA Grand Challenge 2005. [online], Date of visit to site 24.4.2012.
URL http://archive.darpa.mil/grandchallenge05/

[84] DARPA Urban Challenge 2007. [online], Date of visit to site 24.4.2012.
URL http://archive.darpa.mil/grandchallenge/

[85] What we're driving at; blogpost by Sebastian Thrun. [online], Date of visit to site 24.4.2012.
URL http://googleblog.blogspot.com/2010/10/what-were-driving-at.html

[86] Thrun, S.; Montemerlo, M.; Dahlkamp, H.; et al.: Winning the DARPA Grand Challenge. *Journal of Field Robotics*, 2006, accepted for publication.

[87] Urmson, C.; Anhalt, J.; Bagnell, D.; et al.: Autonomous driving in urban environments: Boss and the Urban Challenge. *J. Field Robotics*, volume 25, nr. 8, 2008: pages 425–466.

[88] Viola, P.; Jones, M.: Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2001*, volume 1, nr. C, 2001: pages I–511–I–518.

[89] Wang, Z.; Fan, B.; Wu, F.: Local Intensity Order Pattern for feature description. In *ICCV*, 2011.

[90] Wedel, A.; Pock, T.; Zach, C.; et al.: An Improved Algorithm for TV-L1 Optical Flow. *Statistical and Geometrical Approaches to Visual Motion Analysis: International Dagstuhl Seminar*, 2009: pages 23–45.

[91] Wojek, C.; Schiele, B.: A Dynamic Conditional Random Field Model for Joint Labeling of Object and Scene Classes. In *European Conference on Computer Vision (ECCV)*, 2008.

[92] Wu, Q.; Zhang, W.; Kumar, B. V. K. V.: Example-based clear path detection assisted by vanishing point estimation. In *ICRA*, IEEE, 2011, pages 1615–1620.

[93] Xiao, J.; Quan, L.: Multiple View Semantic Segmentation for Street View Images. In *International Conference on Computer Vision (ICCV)*, 2009.

[94] Xiong, X.; Munoz, D.; Bagnell, J. A.; et al.: 3-D Scene Analysis via Sequenced Predictions over Points and Regions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[95] Zalud, L.: *Robocup 2003: Robot Soccer World Cup VII*, chapter Rescue Robot League - 1st Place Award Winner. Springer-Verlag, 2004, ISBN 3-540-22443-2.

[96] Zalud, L.: Orpheus - Reconnaissance Teleoperated Robotic System. In *16th IFAC World Congress, Prague, Czech Republic*, 2005.

# LIST OF SYMBOLS, PHYSICAL CONSTANTS AND ABBREVIATIONS

| | |
|---|---|
| $\mathbf{M}$ | Matrix |
| $\mathcal{F}$ | Set or operator |
| $\mathbf{v}$ | $n$–dimensional vector |
| $\mathbb{R}^d$ | $d$–dimensional vector space |
| $\mathcal{O}$ | Upper bound of the algorithm's running time |
| tr$\mathbf{A}$ | Trace of $\mathbf{A}$ |
| EM | Expectation-Maximization |
| CRF | Conditional Random Field |
| GMM | Gaussian Mixture Model |
| LBP | Local Binary Patterns |
| LDOF | Large Displacement Optical Flow |
| RBF | Radial Basis Function |
| SHIM | Stacked Hierarchical Inference Machine |
| UGV | Unmanned Ground Vehicle |

# LIST OF APPENDICES

# A NAIVE APPROACHES

In fact, this section does not give any important information about reaching the goal and can be skip by the reader, however I decided to write this section to illustrate the difficulty of the task since almost all computer vision application use some kind of feature matching and one might think, that feature matching is not an issue anymore. Let us mention widely-used panorama stitching, image based localization and retrieval for all of them; however only distinctive features are matched in these applications and even with such constraint only a relatively small subset of them are matched correctly, while we need to find the correspondences for every single pixel in an image sequence. The other reason is to summarize some of the naive approaches which we have tried to use and concluded, that all of them can be considered as doomed for this task.

## A.1 Averaging over the Time Dimension

We start our journey through the land of no hope with simple averaging over the pixels at the same spatial coordinates at each frame of the time window. This is commonly used technique in the domain of 1D signal processing, especially in biomedical engineering applications.

And for static scene, it really works great (Fig. A.1 (a). Unfortunately, we are interested in dynamic scenes, and as we have already mentioned in the previous section, (almost) everything is moving somehow in that case, which means, that there are either a big lag between predictions and correct labeling, and for some pixels we average pixels that totally do not correspond (Fig. A.1 (b).

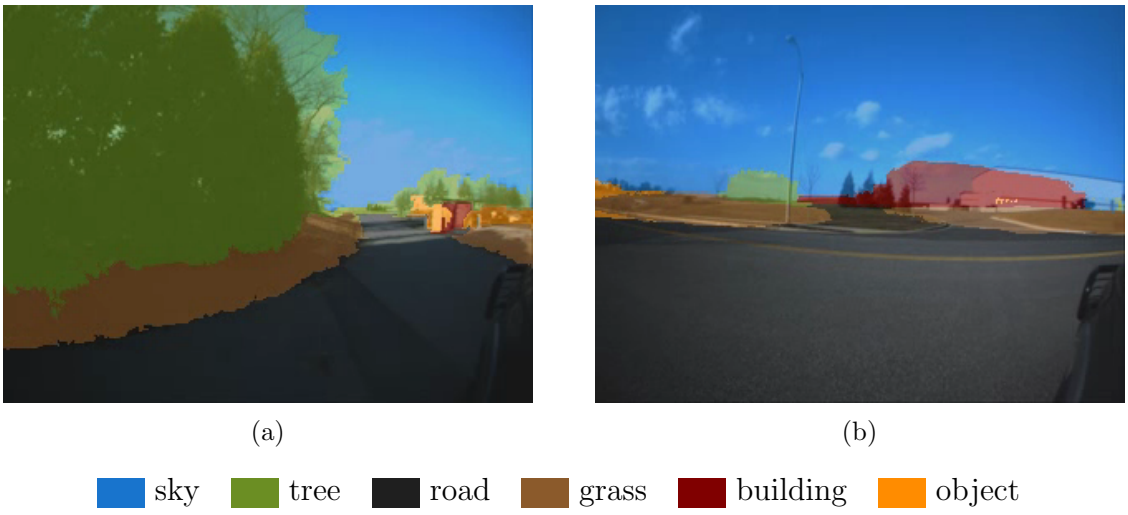



Fig. A.1: **Averaging over the time dimension**: time window consists of 10 frames, (a) static scene, (b) failures in dynamic scene caused by large motion.

## A.2 Extended FH-segmentation

FH segmentation is a popular algorithm proposed by Felzenszwalb and Huttenlocher [20] for segmenting of an image into regions. Its popularity comes from the following properties: (1) ability to preserve detail in low-variability image regions while ignoring detail in high-variability regions, (2) nearly linear time complexity with the number of graph edges.

In fact, even more important is, that all segmentation methods fail in some situations and because FH segmenter is based on graph-based approach, it is very efficient and usually is used to obtain over-segmented regions, that are used as inputs to some more advanced algorithm. Obviously, this is reasonable, since it is always better to preserve region boundaries, even if it means, that the image is over-segmented.

FH-segmenter builds a graph, where vertices correspond to the image pixels and edges are weighted by color distances between them. The algorithm is closely related to Kruskal's algorithm for constructing the minimum spanning tree of a graph, that can be implemented to run in $\mathcal{O}(m \log m)$, where $m$ is the number of edges in the graph using a disjoint-set forest with union by rank and path compression.

As we have mentioned in the state-of-the-art, FH extension into time dimension is nowadays popular in the domain of action recognition [60], [47]. However, several reasons exist why this approach is not suitable:

1. Extension into time dimension is not straightforward – it is possible to define fully connected graph or graph which is fully connected just in some frame and sparsely connected with other frames. Unfortunately, it might happen, that there appears a cut, which links pixels from completely different semantic regions.

2. It is possible to define the graph on pixels or regions, however in the latter case it is difficult to define the graph, since it is necessary to find the overlapping regions (this might be overcome with "intersection matrix"). Moreover, although the regions provide a better support for determining of weights (it is possible to use e.g. $\chi^2$ distance between histograms), it might happen that two regions from different semantic class have lower weight in some frame than all others.

3. Obviously, we loose one of the biggest advantage of stacked hierarchical inference machines (and other methods), that implicitly expects, that the segmen-

tation fails and robust to these situation. In that case, we run segmentation again, which might completely destroy even correctly labeled regions, instead of performing of a simple smoothing.

4. It is necessary to perform the segmentation in a moving window (in time) and thus we need somehow weight the frames in the front and end of the window to ensure smooth transition between the windows.

5. Algorithm has very high memory complexity.

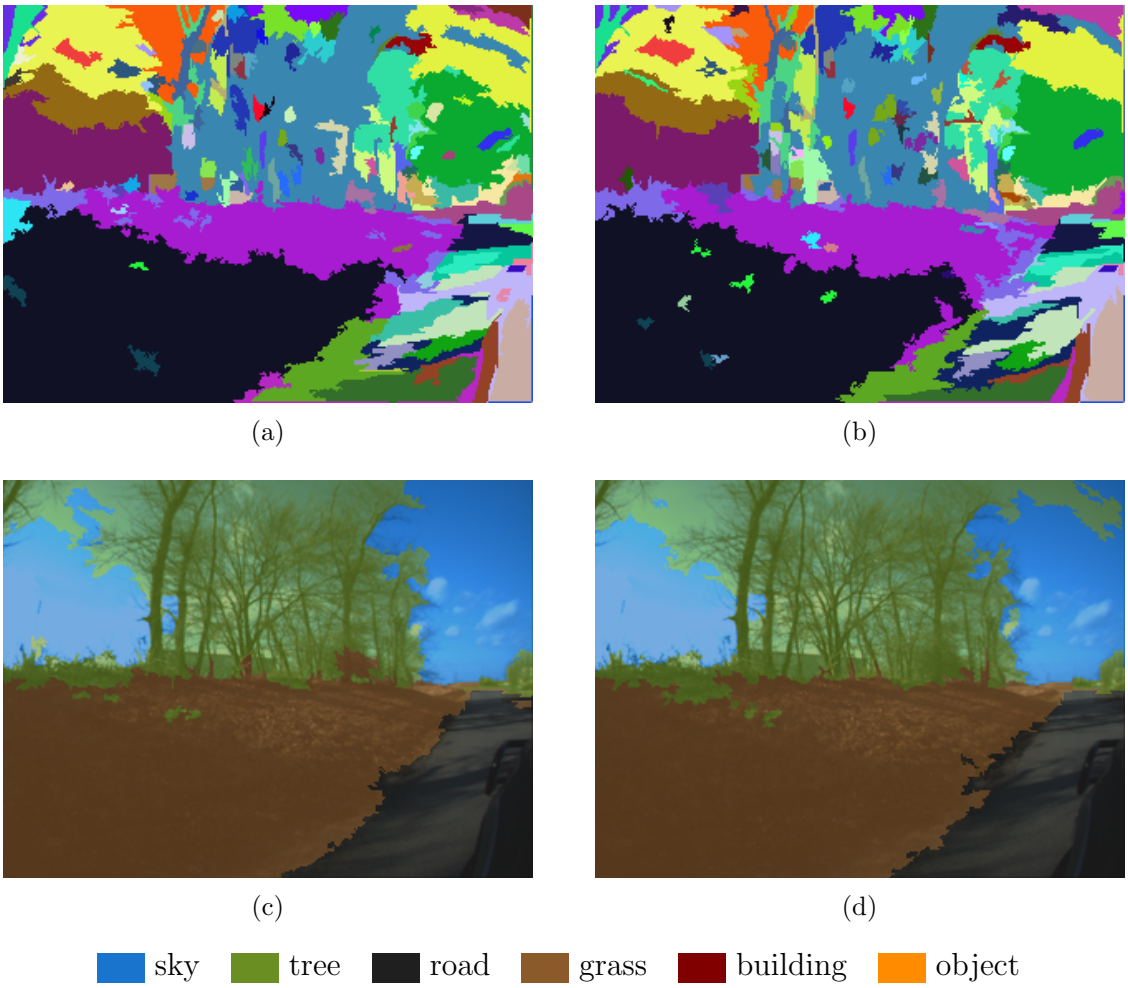



Fig. A.2: **FH segmentation extended to time dimension**: defined over the regions – (a) and (b) are segmented regions in pseudocolor (corresponding regions are in the same color), while (c) and (d) are output frames – (leakage of tree and grass classes).

## A.3 Feature Matching

The last example we show in this section is closely connected with matching of local invariant features, widely used in instance-based recognition applications such as automatic panorama stitching.

One of the most influencing papers in computer vision is the seminal work on SIFT [52]. During the past decade, a variety of improvements were proposed [3], [72], [13], [46], [73, 89]. Such descriptors were reported to improve the efficiency and matching accuracy upon SIFT. However, it can be shown, that the performance of matching of pixels instead of keypoints is still too low to be used in a global setup.

To demonstrate this, we report the performance of the state-of-the-art descriptors such as SIFT and SURF compared with recently proposed LIOP, MROGH and MRRID. Since our application is time-constrained and real-valued descriptors need to be matched in a multiple randomized kd-tree setup ($N$ trees, $\epsilon$-*approximate* nearest neighbor search), we add to the evaluation alternative descriptors represented by binary descriptors BRIEF, BRISK and ORB.

We follow the evaluation protocol from [55], which is based on a number of correct and false matches obtained for a given image pair. However, we evaluate the matching performance in a database of features that also contain distractors, that are features from different images. We use the precision-recall score in this section.

We do not discuss the results in detail here, since it is obvious, that the overall performance is too low to be useful for matching of pixels instead of keypoints (results are presented for distinctive keypoints). More detailed performance evaluation including repeatibility score, speed-up and detectors/descriptors efficiency is given in our paper [56], however this is beyond the scope of the thesis.

Tab. A.1: Precision/Recall for the different descriptors and $N = 40$, $e = 3$

| Detector | Descriptor | Precision | Recall | MAP |
|---|---|---|---|---|
| SURF | SURF | 0.2956 | 0.3352 | 0.3338 |
| SURF | SIFT | 0.3268 | 0.4111 | 0.4913 |
| SURF | BRIEF | 0.3557 | 0.3946 | 0.5136 |
| SURF | ORB | 0.2612 | 0.2686 | 0.4374 |
| SURF | LIOP | 0.5419 | 0.6969 | 0.5683 |
| SURF | MROGH | 0.5475 | 0.6798 | 0.5274 |
| SURF | MRRID | 0.5620 | 0.6882 | 0.5098 |
| SURF | BRISK | 0.3970 | 0.4641 | 0.5298 |
| BRISK | BRISK | 0.3056 | 0.3771 | 0.4915 |
| ORB | ORB | 0.2468 | 0.2654 | 0.4632 |

# B ENCLOSED DVD

Enclosed DVD contains the following directories:

- `Thesis`
- `Video`
- `Code`